MODELING THE MERGER OF THE CLASSIFIED NETWORKS OF THE DDN: BLACKER

THESIS

Roger Swope
Captain, USAF

AFIT/GOR/ENS/88D-21

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/88D-21

(1)

MODELING THE MERGER OF THE
CLASSIFIED NETWORKS OF THE DDN:
BLACKER

THESIS

Roger Swope
Captain, USAF

AFIT/GOR/ENS/88D-21

AFIT/GOR/ENS/88D-21

MODELING THE MERGER OF THE CLASSIFIED NETWORKS OF THE DDN: BLACKER

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Operations Research

Roger Swope, B.S.

Captain, USAF

November 1988

Approved for public release; distribution unlimited

Acknowledgements

I have had the good fortune of pursuing a topic that I really enjoyed working on. Consequently, much effort and many long hours spent working on this product did not seem like work.

I owe a debt of gratitude and thanks to many individuals. First, I wish to thank my thesis advisor, Major Kenneth Bauer. His contributions and assistance in finalizing this research effort and maintaining a proper perspective throughout were greatly appreciated. Second, I thank Major Joseph Litko, my reader, for his inputs to the final product. I am also indebted to Mr Terry Damon, B651 DCA, for his guidance and assistance in providing and narrowing the research topic. Thanks to all those individuals at the DCA and NSA who gave of their time to provide answers to my never ending series of questions.

Finally, I wish to thank my family for their patience and support. And last, I thank my beautiful wife Mary Ann for the sacrifices made and the encouragement given during the past 18 months at AFIT.

Roger L Swope

Table of Contents

List of Figures

List of Tables

List of Acronyms

ACC	Access Control Center
ACK	Acknowledgement
ADP	Automatic Data Processing
ANSI	American National Standard Institute
ARPANET	Advanced Research Project Agency NETwork
BBN	Bolt Beranek and Newman
BIC	BLACKER Initiation Carrier
BFE	BLACKER Front End
BLACKER	NSA Program to Produce E3 Communication Network Security Devices
COI	Community of Interest
DARPA	Defense Advanced Research Projects Agency
DCA	Defense Communications Agency
DDN	Defense Data Network
DISNET	Defense Integrated Secure Network
DoD	Department of Defense
E3	End-to-End Encryption
ICD	Interface Control Document
IEEE	Institute for Electrical and Electronic Engineers
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
IST	InterSwitching Trunk
KDC	Key Distribution Center
LAN	Local Area Network
LLD	Logical Link Disconnect
LLDA	Logical Link Disconnect ACK
LLR	Logical Link Request
LLRA	Logical Link Request ACK
MILNET	Military Network
MINET	Movement Information Network
MTBF	Mean Time Between Failures
NSA	National Security Agency
OSI	Open Systems Interconnection
PB	Plackett-Burman
PSN	Packet Switching Node
SACDIN	Strategic Air Command Digital Network
SAS	Software System for Data Analysis Registered Trademark of SAS Institute Inc
SCINET	Sensitive Compartmented Information Network
SLAM II	Simulation Language for Alternative Modeling Registered Trademark of Pritsker & Associates
SPF	Shortest Path First
WINCS	WWMCCS Intercomputer Network Communication Subsystem
WWMCCS	World Wide Military Command and Control System

Abstract

The purpose of this thesis was twofold; first, to determine the impacts of adding the new E3 device, BLACKER, to a network of the DDN and second, to develop a simulation model that would lay a foundation for modeling the merger of the DDN. A SLAM II computer simulation model was developed to simulate two networks of the DDN. Components of the BLACKER system were added to the networks. After BLACKER was installed, the two networks were merged into a single Segment.

Sensitivity analysis was performed on the networks input parameters. Analysis was also performed on the model's output parameters (values of the SLAM II Output Report) to determine the impacts of the BLACKER system on the performance of the individual networks and the Segment.

MODELING THE MERGER OF THE CLASSIFIED NETWORKS OF THE DDN: BLACKER

I. Introduction

Background

The Defense Data Network (DDN) is a packet switching network designed to support the long-haul data traffic of the military services and defense agencies, both in the United States and overseas. The DDN was created in 1982 based upon the technology and architecture of the Advanced Research Projects Agency Network (ARPANET). The ARPANET, funded through a research and development program sponsored by the Defense Advanced Research Projects Agency (DARPA), began as an experimental packet switching host-to-host network in 1969. The ARPANET was to advance computer networking by providing efficient communications between heterogeneous computers and networks allowing them to share hardware, software, and data resources. (DCA, 1985:21) The ARPANET was designed to include a var'ety of geographically dispersed users. (DCA, 1986:2-15)

The DDN, based on ARPANET packet switching technology, meets the intercomputer telecommunications needs of the Department of Defense (DoD) and satisfies the DoD worldwide wartime survivability requirements. The DDN was designed

and implemented with survivability, security, and interoperability as essential goals. (DCA, 1985:3-24)

The DDN is required to supply both the long-haul and local area data communications services to DoD operational systems including the Worldwide Military Command and Control System (WWMCCS), intelligence systems, general purpose Automatic Data Processing (ADP) systems, and other command based systems and data networks. The DDN is required to have the flexibility to accommodate significant changes in ADP and data communications technology, in user requirements, and in any significant economic factors that could influence the program. (MITRE, 1984:1-34) In order to satisfy these requirements, the DDN must:

1. provide interoperability among heterogeneous devices;
2. provide for multi-level secure data communications;
3. provide reliable data transmission;
4. provide a high level of availability;
5. provide for precedence and preemption;
6. maximize the use of DDN standardized components;
7. use low risk technologies;
8. be easily expandable. (MITRE, 1984:2-19)

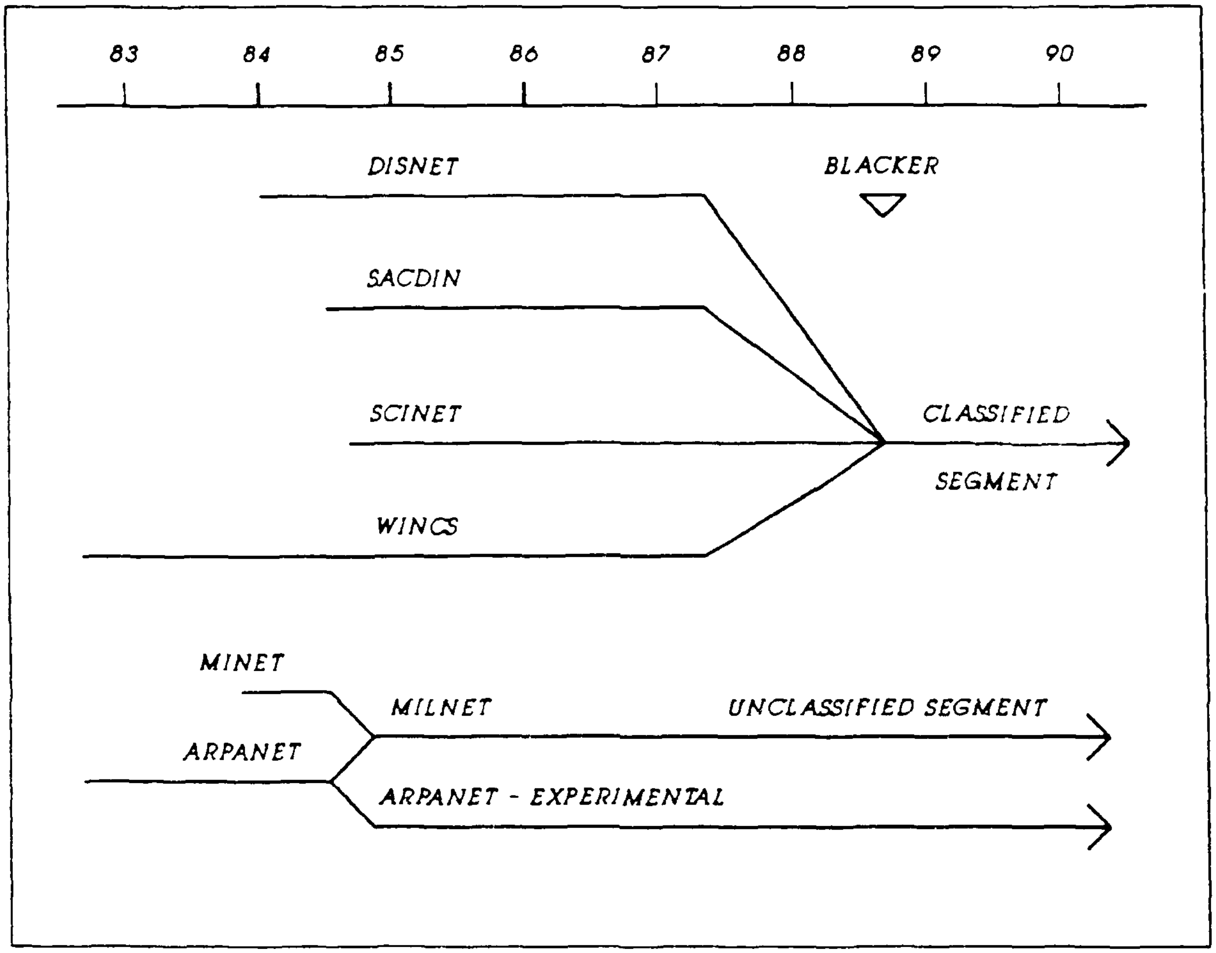
The packet switching communications technique used by the DDN is accomplished by interconnecting trunk circuits and packet switching nodes that form a communication backbone. Message traffic entering a host computer from a terminal is processed in standard length message segments. These segments or packets are transmitted from switching node to switching node over the backbone until they reach their destination node. At the destination switching node, they

are retrieved and restored in the form of the original message for delivery to the destination.

Multiple routes are available over various links of the network. Each DDN packet switching node (PSN) is connected to several other switching nodes of the same network by dedicated interswitching trunks (ISTs). Each PSN, attempting to deliver a packet to its destination in the shortest time, has the capability to route that packet over any of the PSN's connected trunks. Thus packets of the same message need not cross the backbone over the same links. If a switching node or trunk fails, the network automatically reroutes data packets over alternate paths toward their destinations. The same technique is employed to route traffic around busy or congested areas. The routing technique and resulting paths for each packet of a message are transparent to the message originator and receiver. Each link of the network can be utilized by many packets of different messages or sources, allowing many users access to each IST virtually at the same time. The DDN is designed for continuous operation to provide a user availability of 99% or better.

The DDN has developed in evolutionary steps. It will continue to evolve into a complex dispersed system by combining several existing networks into a central unified network with a common communication backbone. (Selvaggi, 1982:111) This evolution is illustrated in Figure 1,

concept taken from the 'The DDN Course' instruction manual. (DeVere and Passmore, 1986:Ch 5, 2)



(DeVere and Passmore, 1986:Ch 5, 2a)

Figure 1. DDN Schedule

The initial stage was based on the merger of unclassified networks; the ARPANET and the Movement Information Network (MINET) combined to form the Military Network (MILNET). Further DDN development has included the establishment of separate networks for different security levels. The DDN

currently is composed of the classified and the unclassified segments. (Fidelman, 1986:155) The unclassified segment consists of two unclassified networks, and the classified segment has four classified networks. The classified networks each contain a separate single level secure system, referenced as separate communities of interest (COI). These five baseline systems will expand to support any additional DDN system requirement during the transition to the unified DDN. Currently, the classified segment is using link-to-link encryption devices to protect the transmitted data. The DDN will remain a segmented network until installation of the new BLACKER devices which will allow the four classified systems to combine under a single classified segment. Then the two segments of the DDN can be combined into the unified network. (DCA, 1985:1-25)

The BLACKER Front End (BFE), a new end-to-end encryption (E3) device, will function as a front-end processor between the existing host computer and its packet switching node. The BLACKER Access Control Center (ACC) will monitor, control, and audit system security features. The ACC will authorize host-to-host secure connections. (Messah, 1987) The Blacker Key Distribution Center (KDC) will provide automatic key variable generation and distribution to the BFEs within its COI. The components of the BLACKER system, acting together will provide a multilevel secure environment suitable for all DDN systems. (MITRE, 1984:1-7)

Problem Statement

The purpose of the thesis research was to determine what impacts BLACKER would have on the existing capabilities of a network of the DDN, and to investigate the effects of the network transition as the network integrates into the unified segment. A BFE must be installed on each host of the network before the network can be merged into the segment.

The Defense Communications Agency (DCA) does not have a model to study network or segment characteristics after the BFE is installed on each host of the network. (Damon, 1988) The thesis research addressed the effects of BLACKER on the following characteristics of a single segment (that consists of 2 networks) of the DDN:

1. Packet and message throughput.
2. Host throughput and input limitations.
3. Packet and message delays.
4. System bottlenecks, host or node congestion.

DCA is currently aware of some advertised system constraints and limitations. The BLACKER system will add approximately 100 milliseconds to packet delay, according to a study performed by the SPARTA, INC. (SPARTA, 1985:54) The effect of the BLACKER system will not fully be realized until implementation is complete. DCA would like a model to study the projected impact of the BLACKER devices on the host, on a network, and on the combined segment. (Damon, 1988)

Scope

To develop a large scale in-depth model of the Defense Data Network is beyond the scope of this research. This investigation was limited to a single segment of the DDN. Areas of concern are listed above, emphasis was placed on system delays and message throughput.

The scope of this research was to develop a simulation model of a network of the DDN, to investigate variations in input parameters on the model, and to analyze the influence of BLACKER on that network as it integrated into the unified segment. (Damon, 1988) SLAM II (Pritsker, 1986), an advanced FORTRAN based simulation language, was used to build the network model. This investigation was limited to the performance of two closely related networks of the same segment. The segment under investigation consisted only of these two networks. The approach involved in this thesis does not necessarily predict the performance of any other particular network. Rather, it is intended to develop a mechanism for relative comparisons between these two networks in the pre- and post-blacker environment.

This research does not consider how to merge the two networks or the best implementation scheme. The research does not address the introduction of additional links (ISTs) or additional switching nodes (PSNs).

Research Objective

The research objective was to create and analyze a computer simulation model of a segment of the DDN. The segment consisted of two generic networks. A model was developed of each network and examined in a steady-state condition. Statistics obtained for the model prior to BLACKER were validated with actual data obtained from DCA and the National Security Agency (NSA). The model was modified by including the BLACKER device on all host computers in the networks. The additional system requirements of BLACKER were added to the model. Statistics obtained from the modified model were compared to those obtained prior to the BLACKER device.

Assumptions

The BLACKER device is currently under test and evaluation by the NSA and the DCA. Characteristics noted of the devices tested were assumed to be true for the entire BLACKER population.

Traffic data collected prior to BLACKER was assumed to be characteristic of the traffic condition after BLACKER is installed.

The research did not study the survivability of the baseline system, the segment, or the overall DDN network. The network survivability has been enhanced by the addition of each new node and link. The inclusion of a system or network within the existing backbone can only make the

segment more reliable and survivable. (Heidi, 1982:61) The BLACKER will have no adverse effect on system survivability. Network survivability will not be considered in the scope of this research.

The BLACKER device is not intended to degrade the network in any performance parameter. The BLACKER is intended to enhance the existing system. The estimated system performance in a post-BLACKER state should be equal or superior to the performance prior to BLACKER. Simulated failure of the BLACKER device was not included in the model. BLACKER has an estimated mean time between failures (MTBF) of at least 4000 hours. (Messah, 1987:4) DCA felt this MTBF was not significant enough to be addressed in the time simulated during these studies. (Damon, 1988)

Overview

Chapter Two presents a summary of current research and documentation on related studies and issues. Chapter Three presents the methodology and the model development. Chapter Four presents the design of the experiments to determine model sensitivity and the results of the model simulation for each network and the combined segment. Chapter Five summarizes the thesis and presents the final results. Chapter Five also makes recommendations for future research.

II. Literature Review

Method of Organization

There are many factors against which network performance can be evaluated. Recent performance evaluations have studied the effects of varying input traffic, PSN processing time, and both PSN and IST overhead. Studies have addressed link and network reliability, decomposition strategies (from a survivability view point), and interconnection network processing. There is an extensive amount of literature available on packet switching and communications networks. The literature reviewed that pertained to this study of the DDN is listed below in a topical order: DDN management and operations, descriptive model simulations, model parameter screening techniques, and routing algorithm designs.

Within this document, references to BLACKER will include all components of the BLACKER system: BFEs, ACC, and KDC.

DDN Technical Literature

DCAC 310-P70-X, DDN User Operating Procedures, provided a comprehensive description of the DDN and formalized operations procedures for the various network services. The basic security architecture currently used until BLACKER is implemented is outlined in the Defense Data Network Subscriber Security Guide. Additional Security Requirements for the Next Generation Packet Switch provided those additional security features that were required after

BLACKER implementation but were not included in the guide above. It included special interpretations for the DDN, both with and without BLACKER. The DDN Protocol Handbook, Volumes 1, 2, and 3 served as a guide for subscribers interested in implementing the DoD suite of protocols on their computer host or terminal to be attached to the DDN.

The BLACKER Interface Control Document (ICD) defines the interfaces for the BLACKER Front End (BFE) processor. The BFE requirements are specified in the Device Functional Specifications for the BFE Device, Document No. B002-20. The BLACKER Test Plan, Technical Report 7713-TR-01, provides guidance for the technical effort to be used throughout the test period and establishes the criteria necessary for BLACKER evaluation. Document No. B002-21-R04, Device Functional Specification ACC BLACKER Program specifies the performance, design, development, and test requirements for the BLACKER Access Control Center (ACC). Messeh's working papers (1987) on the DISNET BLACKER Domains addresses the structuring of the DISNET system into various domains and the selection of the appropriate ACC and Key Distribution Centers (KDC) for each domain. Messeh identifies the number of domains required for each security level and proposes a design configuration to satisfy the required system reliability and survivability.

Numerous other DoD documents and military specifications reference various aspects of the BLACKER program and the DDN in general. Only a few were listed here.

Descriptive Models

Simulation models are available to simulate various aspects of any Integrated Services Digital Network (ISDN) or Local Area Network (LAN). The models vary in size from Patel's "Design of a Small Packet Switching Node" (1986) to Hofstetter's "Traffic Models for large ISDN-PABX's" (1985). The models deal with integrating a single host into an existing network (Patel, 1985) to interconnecting huge networks into a super structure. (Narayanan, 1987) Models have been developed to simulate methods of congestion control, stability, adaptive routing, and primary path determination.

Models of concern for this study must be developed in the guidelines established for the Reference Model of the Open Systems Interconnection (OSI). The Reference Model is an International Standard that provides a conceptual and functional framework for developing sublevels of the OSI architecture. (ANSI, 1981:82) The Reference Model presents a logical view of interconnecting computer systems, and refers to standards for exchanging information among terminal devices, computers, and networks. (ANSI, 1981:83) This international network design is embedded in the basic DDN concept.

Labetoulle and Pujolle in 1981 developed several models to compute the performance of computer networks based upon the OSI recommendations. They developed a model with 7 terminals and 4 hosts. The model represented a queueing network that used internal queues for each data link. Labetoulle studied the response times and determined system stability based upon system window size. The window size specifies the number of packets that may be transmitted from point A to point B without being acknowledged. The study concluded that a small window width should be selected; a large window does not allow for more throughput or better response time. (Labetoulle and Pujolle, 1981:126)

Gerla and Chan (1985) formulated the optimal window selection assignment as a mathematical programming problem, and showed that exact solution is computationally impossible. As window size increases, throughput increases; since more packets are allowed in the network, queueing delays also increased. This delay has a feedback effect on throughput and the window size must be adjusted. Gerla and Chan confirmed the results of the previous study by Labetoulle and Pujolle. Meditch and Lea (1985) developed a different approach for the stability and throughput analysis in the forms of inequalities and formulated a model using an embedded Markov chain. Tcha and Maruyama (1985) developed an algorithm that takes a straightforward iterative approach to find the selection of primary path on a communication

network, a concept similar to the Simplex method. Rosenberg developed a "nonlinear programming heuristic for computing optimal link capacities in a multi-hour alternate routing" to determine the optimal number of communications channels required in a network. (Rosenberg, 1987:354-356)

Tropper (1987) examined the use of priorities in the network to speed the return of acknowledgements and expanded the study to effect a grade of service routing. Thus, shorter messages are given preferential treatment. Tropper employed a queueing network model of the communication subnet and used a "heuristic mean-value analysis algorithm as our principle computational tool." (Tropper, 1987:89) Narayanan (1987) presented an interworking model to combine networks based on interfacing elements or gateways. The model provided transparent end-to-end connections between users of the networks that were connected. Cerf and Cain (1983) presented "The DoD Internet Architecture Model" approach where a common internal host-to-host packet or datagram service is proposed for interworking the networks. Both techniques above do not encounter the problem where the networks involved have different length packets. Bennett (1982) addressed this problem when the joining networks have different size packets, and developed a model to determine fragmentation (the act of breaking larger packets into smaller ones allowing them to transverse intermediate networks that support a smaller maximum packet) overhead.

Model Parameters

System parameters required for the model were provided by NSA from the BLACKER network tests and by DCA for control data from an existing network database. The parameters under study were based on the following input message distributions: input source-destination, input packets per message, and bits per packet. Network 1 was modeled to conform to a database constructed from the 24 July 87 Quarterly Status Report R4317; this network had 17 nodes and 25 links. Network 2 was constructed with 10 nodes and 14 links; the input distributions were evenly distributed over the network. See Appendix D for the databases used in the various network designs. The models developed were based on a priority queuing structure similar to those discussed above. The baseline model was a simulation similar to the model of "The DoD Internet Architecture Model" presented by Cerf and Cain. (1983)

Parameters for system specifications and requirements not previously measured were extracted from the appropriate interface document. For example, the BLACKER ICD was used to provide processing time and delay within the BFE.

The data distributions above were varied on the networks to determine the effect on the system response. Since the distributions were estimates for a given time period, model sensitivity was determined by simulating experimental runs with perturbations of these inputs. Using the techniques

discussed below, a regression model was developed to indicate the most significant parameters of each network. The original design had 89 variables under investigation. The supersaturated simulation model was reduced by group screening and modified group screening techniques, see Chapter IV for details. The following literature review of screening techniques will contain examples with data referencing the actual design and development used in this thesis effort.

Supersaturated Screening Designs (Mauro, 1986)

The article "Efficient Identification of Important Factors in Large Scale Simulations" by Carl A. Mauro (1986) presents a description and a brief discussion of seven different types of factor screening experiments when dealing with supersaturated designs. These techniques are extremely useful when the model under investigation is large or complex with many variables or parameters, either input or output. In a supersaturated situation, there are more factors or parameters in the model than runs or responses available for screening the model. Usually there are only a selected number of parameters that have a major impact on the model. Thus, it become advantageous to have a technique to identify the major factors or the most important players in the model. After the unimportant factors are identified and eliminated from the model, then time and resources can

be concentrated on studying the effects of the major players.

The seven designs considered are:

1) random balance,
2) systematic supersaturated,
3) group screening,
4) modified group screening,
5) T-optimal,
6) R-optimal,
7) and search designs. (Mauro, 1986:296-304)

Mauro advised not to ignore prior information that is known about factors of the model. In some cases, screening can be performed at an early stage to reduce the number of variables under investigation. Important factors that are known can be held constant to observe the questionable effect of factors not yet grouped as important or unimportant. Unimportant variables can be ignored to determine the major effects and interactions of the important factors. The main function of factor screening is to group the factors either as important and requiring further investigation or to determine the factors unimportant and eliminate them from further consideration.

Initially screening can be used to determine the factors that have major effects by estimating the first order model where the response r_i is the result of the i-th simulation run.

$$r_i = B_0 + B_1X_1 + B_2X_2 + \dots + B_nX_n + e_i \qquad (1)$$

N is the total number of factors that were evaluated at each normalized level with coded values -1 and +1. X_i is

the level of the n-th factor during the ith simulation run. B_0, the mean response, is that component common to all observations or simulation responses. B_i is the coefficient of the level of the n-th factor. The random error, e_i, has mean 0 and some unknown variance. The coded value is obtained by a convenient linear transformation of the original parameters of interest. (Box and Draper, 1987) The model can be useful over a relatively small region of the response surface, and may not be ideal (or even meaningful) for use outside this small region. The range of the coded values could be used to determine if the model is sensitive to variations of that parameters within a selected delta, thus the model could be used as a form of sensitivity analysis.

Random Balance Designs. In a two-level random balance design, each factor is ranged over the -1/+1 coded value in a random fashion. The main requirement is the total number of runs N is an even number, where each factor (column of the design matrix) has an equal number of coded high and low values. These coded values need not have a pattern or sequence. Thus all possible combinations of N/2 +1's and N/2 -1's are randomly selected. Consider the example given in Table 1 with a limited number of the variables shown. Ten run were recorded with 19 variables randomly ranging high to low in their coded values. Note that no two runs have the same values across the rows.

Table 1. Random Balance Design

run nbr \ variable name	A1	A2	A3	A3	A5	A6	A7	A8	....	A19
1	-1	-1	-1	-1	-1	-1	-1	-1	...	1
2	-1	-1	-1	-1	-1	-1	-1	1	...	-1
3	-1	-1	-1	-1	1	1	1	-1	...	1
4	1	1	1	1	-1	-1	1	1	...	-1
5	1	1	1	-1	1	1	-1	1	...	-1
6	1	-1	-1	1	1	1	1	-1	...	-1
7	-1	1	1	1	-1	1	-1	-1	...	1
8	-1	1	1	1	1	-1	1	-1	...	1
9	1	-1	1	-1	1	-1	1	1	...	-1
10	1	1	-1	1	-1	1	-1	1	...	1

Mauro sites flexibility as a principal advantage of this method; this design is extremely easy to prepare for any number of parameters and any even N. Also the number of runs can be selected independently of the number of variables. The only restriction is stated above; N must be even and the variables must have an equal number of high/low settings. The fact that the parameter are randomly set makes this design more desireable when the number of parameters is large.

The advantages can also cause disadvantages. Randomly selecting the parameters can also cause confounding to be random. Analysis is also an area of concern, there is no generally established criteria or method. Mauro suggested to consider each factor separately and apply the F-test or some other standard analysis technique (similar to least squares stepwise or stagewise regression).

Considering the F-test on an individual basis, a scaled version of this design was performed on the model used for Network 1. A stepwise regression was performed (using SAS) with an F-value set to .15, the following parameters listed in Table 2 entered the model in the order given:

Table 2. Example of Stepwise Regression

step	variable name	F-value	PROB>F
1	A8	24.54	.0001
2	A15	7.45	.0143
3	A14	4.39	.0524
4	A12	5.40	.0346
5	A7	6.65	.0219
6	A2	7.19	.0189
7	A1	3.30	.0945

where A1 is the input traffic distribution for PSN1,
A2 is the input traffic distribution for PSN2, ... etc

A single F-test was performed on the model r_i = A18, a parameter that is known to be non-significant. A18 is the input traffic distribution for PSN18. The F-value for this model was 0.269 with a PROB>F of 0.6085. On an individual basis, Mauro questions what F-value is significant. Draper and Smith (1981) recommend setting the F-value at some threshold and treat only those values above as significant (also identical to the concept used in stepwise). Mauro states that the random block design would be "inefficient for detecting all but the very large effects."

Systematic Supersaturated Designs. This concept is very close to the random block designs with the additional requirement to "systematically attempt to minimize confounding." The systematic supersaturated designs, constructed by Booth and Cox (1962), minimizes $\max_{i=j}[c_{ij}]$ where $c_{ij}=\underline{x}_i'\underline{x}_j$. If two or more designs have the same minimax value, then use the design that minimizes the number of pairs of columns to attain the minimax value. Thus, an attempt is made to make the designs as orthogonal as possible.

Booth and Cox (1962) provided the following systematic supersaturated designs for selected combinations (N,K), where N is the total number of runs in the experiment and K is the total number of factors to consider: (12,16), (12,20), (12,24), (18,24), (18,30), (18,36), and (24,30). Designs for values of K ranging between those above (where k is the actual number of factors) can be obtained from the next higher K value design by deleting the final K-k columns.

Main advantage with this design is to minimize the confounding. The major disadvantage is that these designs are not readily available and usually require a computer program to determine the appropriate sequence of experimental runs, especially if the number of parameters in the model is large.

Group Screening Designs. In a group screening design, factors are combined into groups of similar size and that group is then tested as if it were a single factor. Thus a large number of parameters can be combined to reduce the number of experimental runs required. These two-level orthogonal designs require N runs for studying K = N-1 factors. This concept was used on the input parameters of Network 1 to group a 17X8 and a 17X64 table into two 17X1 variables to be used in a Plackett-Burman (PB) design with N=20. Initially the parameters for Network 1 were listed in three tables of sizes 17X17, 17X8, and 17X64. After several initial runs were performed with various values in the latter two tables (values were ranged high and low in their coded values), no significant difference was noted in the response when the first table (17X17) was held constant. The entries were then combined into a single factor for the last two tables to allow the total number of variables to be reduced to 19 (17+1+1). (Box and Draper, 1987:162)

Mauro states that an advantage with this design is the limited control over confounding; factors within the same group are completely confounded and not at all with factors in the other groups. Also grouping the factors together allows for a smaller N and makes the study more manageable. This grouping and testing can be repeated at various stages, each time reducing the number of non-significant factors and coupling them with other non-significants into a smaller

model. This grouping and testing can be repeated for any number of iterations. But those groups designated as significant in previous stages are broken into smaller subgroups to determine their effect in the next series of testing. In the final screening, factors are tested individually.

A disadvantage is that the total number of runs required for the group screening procedure is not fixed. The significant number of group factors carried from one stage to the next can not be determined prior to performing the actual runs. The initial model for Network 1 had 89 variables that reduced to 19 in the first series of runs. A modified group screening techniques (discussed next) was then used to reduce the number of parameters to 7. That model was then reduced by a pure group screening concept to test the significance of the three input parameters only. The second stage required 27 runs, and the final stage required 8 runs. The total grouping and the total number of simulation runs were not known when testing began. Mauro cites a requirement to know some underlying relationship of the model as a second disadvantage. In the DDN model, the underlying relationship was easy to detect based upon 6 runs varying the values of the last 72 parameters to their extremes while holding the first 17 variables constant.

Mauro also suggests that important effects of individual factors may "cancel within the group." Consider two effects

that have opposing relationships and are combined in a group, this grouping may cause the effects to cancel each other out and net no input to the response. If the factor are assigned within the group in such a way that all the factors are working in the same direction, then this disadvantage can be avoided.

Modified Group Screening. In this type of design the total number of runs can be determined prior to any experimentation. The total number of group factors to be tested in the first stage and the total number of factors that will be carried forward to the next series of testing is determined or agreed before testing begins. The model at the end of each stage has the n1 number of most significant effects. Those effects are then tested individually in the next series of tests, and only n2 of them are maintained for the next sequence of screening.

Modified group screening has the same advantage and disadvantages as group screening.

T-Optimal and R-Optimal designs. The T-Optimal design attempts to minimize the trace of $(\underline{X}'\underline{X})^2$, where $\underline{X}$ is the matrix defined as $[\underline{1}, \underline{X}_1, \underline{X}_2, \ldots \underline{X}_K]$. Thus it minimizes the sum of the squared inner products of all pairs of columns in $\underline{X}$ making them nearly orthogonal. Disadvantage for T-optimal designs is that "rules for their general construction have not been developed, nor have any designs been tabulated within the class of two level (± 1) design."

(Mauro, 1986:300) Analysis of this design can be more complex than those previously discussed due to confounding.

The R-Optimal designs attempts to determine the "projection matrix operator onto the space spanned by the rows of $\underline{X}$." R is defined as $\underline{X}'(\underline{XX}')^{-1}\underline{X}$. The diagonal elements range between zero and one since R is a projection matrix. The sum of all the diagonal elements of R are equal to N since the trace of R is equal to its rank. According to Mauro,

> A design is said to be R-Optimal in a given class of designs if it minimizes, over all designs in that class, the maximum diagonal of R. This amounts to making the diagonal elements of R,..., as nearly equal as possible. (Mauro, 1986:302)

The principal disadvantage is similar to that for the T-Optimal; their general performance has not been evaluated. The advantage is that under certain conditions the "posterior variances of the B_i are all equal."

Search Designs. The search design assumes that k of the K factor are non-zero, but the remaining effects are close enough to zero. Thus the technique is to determine which factors are not zero and provide a close estimate to their value. This design has several noted disadvantages; construction of the two level design is difficult, it is assumed that the maximum number of non-negligible effects is known, and the analysis requires a large number of computations even for a small value of N and K.

Routing Algorithms

The algorithm used on the DDN, developed by McQuillan, Richer, and Rosen (1980), is a modified version of the shortest path algorithm originally credited to Dijkstra. Because of it search technique, this algorithm is referenced as the shortest-path first (SPF) algorithm. The basic SPF concept uses a database to describe the network by generating a tree to represent the minimum delay paths from one node to every other node in the network. The database stored at each PSN contains the average delay per packet on every line of the network; it also specifies which nodes are directly connected. The parent PSN determines the average time or delay from itself to all connected PSNs. That average delay is then reported to all other PSNs in the network to ensure that all PSNs have identical routing information available. Thus, all databases in the network should be consistent, within a small time window, and the same calculations made at 'A' will be repeated at 'B', 'G', and 'X' and all other PSNs that a packet encounters on it path from source to destination. (McQuillan and others, 1980:713-715) In this manner a packet will follow a path to its destination avoiding permanent routing loops and areas of heavy congestion.

The SPF algorithm is used to find the shortest paths from a single source node to all other nodes of the network. The algorithm is a step-by-step procedure. First the shortest

path or the closest node to the source is determined. By the k-th step, the shortest path to the k-th node that is closest to the source has been determined. When the shortest path to a node is determined it will be placed in the set N, or according to Mandl the value will be fixed as permanent. (Mandl, 1980:9) During the next step or the (k+1)th step and all following iterations, a new node is added to N or its value marking is changed from temporary to permanent. The distance from this new node to the source is the shortest of the remaining nodes not a member of the set N.

The SPF algorithm initially consists of the root or source node only. The tree is then updated or modified as above to add a new node that is closest to the root and is adjacent to a node already on the tree. (Schwartz and Stern, 1980:265-278) The process continues by iterations until all nodes of the network are added to the tree. The SPF algorithm maintains a LIST of the nodes that have not been added to the tree but are neighbors of nodes that are on the tree. The tree is constructed step-by-step adding the next node with the shortest path. In this manner, all the nodes are added to the tree and the algorithm terminates.

The basic steps of the algorithm are summarized below: (Schwartz and Stern, 1980:267)

1) Initialization. Set N = {1}, the source node. All nodes of the network will eventually be contained in the set

N. Let L(i,j) be the length or cost of the link from node i to j. Define D(x) as the distance from the source node to node x along the shortest path but restricted to using nodes that are within N. For each node x that is adjacent to 1, set D(x)=L(1,x). If x is not adjacent to 1, set D(x)=M, where M is a very large number.

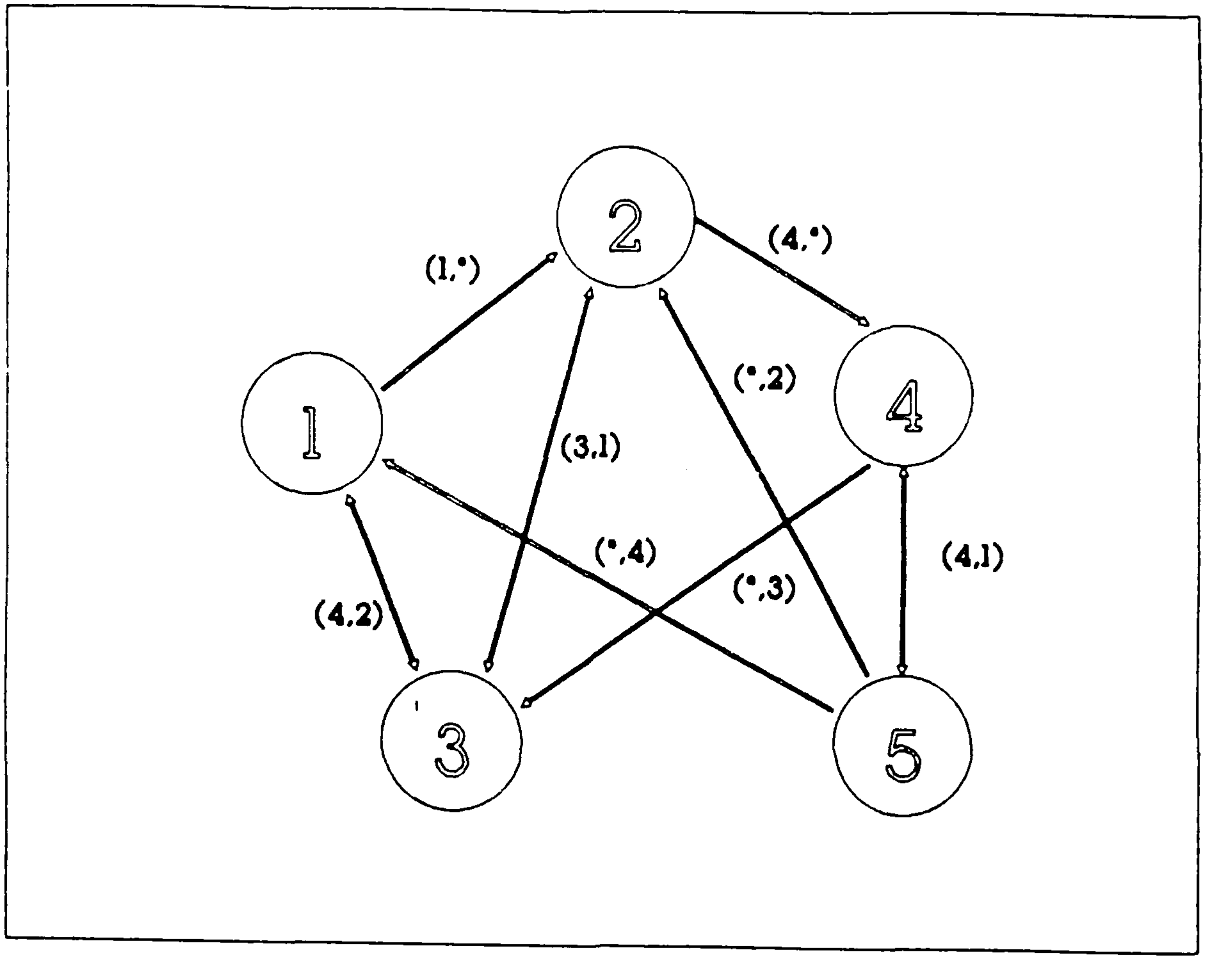
2) Iteration. Find a node w that is not in N but adjacent to a node in the set N for which D(w) is the minimum of all D(x)s, and add w to the set N. Compute the distance for the remaining nodes not in N by

$$D(x)=\mathrm{Min}[D(x),D(w)+L(w,x)] \tag{2}$$

3) Termination. If the sink node is included in the set N (or all nodes are included in N if the minimum tree or the shortest path to all nodes was desired), then terminate. If the sink or destination is not in N, return to step 2.

The procedure determined the shortest path from the source to all nodes of the network. This algorithm must be repeated for each node of the network to determine the minimum path from all nodes to all other network nodes.

Consider Figure 2, an example network provided by Mandl (1979:12). Also note the adjacency matrix for this network in Table 3 where the number associated with the link is the link cost or delay. The above algorithm was applied to the sample network; the resultant matrix of shortest paths and the routing table for node 1 is displayed in Table 4 below.



(Mandl, 1979:12)

Figure 2. Sample Network

In Figure 2, (x,y) represents the arc delay where x is the delay from node i to node j and y is the delay from node j to node i. If the arc between the two nodes does not exist or traffic can not flow in that direction, then x or y is designated by *.

Table 3. Adjacency Matrix for Example Network

node\node	1	2	3	4	5
1	*	1	4	*	*
2	*	*	3	4	*
3	2	1	*	*	5
4	*	*	3	*	4
5	4	2	*	1	*

where * designates that no arc exists from node i to j

(Mandl, 1979:12)

Table 4. SPF - Dijkstra's Algorithm for Example Network

Step	N	D(2)	D(3)	D(4)	D(5)
Initial	{1}	1	4	*	*
1	{1,2}	1	4	5	*
2	{1,2,3}	1	4	5	9
3	{1,2,3,4}	1	4	5	9
4	{1,2,3,4,5}	1	4	5	9

The resulting shortest paths and routing table for any source node of the network could be developed. For example, consider the path from 1 to 2 of the network in Figure 2. Nodes 1 and 2 are connected or adjacent. Node 1 is the only other node in N when 2 enters, Step 2 of Table 4. If the path starts at 1 and the next node is 1 in order to get to node 2, then it is the last node in the path. Consider the path from 1 to 5; note the last row of Table 4. The cost

from 1 to 5 is 9. That value was set in the 5th column when 3 entered N. The path would be 1 to 3 to 5. The journey would be made from node 1 to node 3 at a cost of 4; node 5 can be reached from node 3 at a cost of 5; total travel cost from 1 to 5 is 4+5=9. Any other cost or path can be traced or determined from the above tables in a similar manner. See Chapter III and Appendix E for further development of the routing algorithm.

Numerous other routing algorithms and studies exist. Pirkul (1988) developed a study to address the problem of selecting the primary and secondary route pairs for every pair of communication nodes in a backbone computer network. Pirkul modeled his problem as a mathematical programming problem. Haimo and others (1987) developed in depth tools and testing methodologies for multipath routing algorithms to be compared against the performance of single-path strategies similar to the SPF concept. BBN Report 6195 explored various possible algorithms and techniques of path generation. The intent for this research was to model the existing SPF algorithm used in the DDN. Considering the characteristics of the networks involved, a version of Dijkstra's algorithm or a modified version of Floyd's algorithm would best fit the requirements for this model. (Mandl, 1979:10-14)

Conclusion

This literature review discussed several of the current simulation models and techniques used to model various aspects of a resource sharing network specified in terms of the Reference Model of the Open Systems Interconnection. Numerous models on ISDN characteristics exist, and address a wide range of topics of interest. Only a few of these models are referenced above. This review was not intended to be a comprehensive study of current models and techniques available. Research revealed that each model developed was intended for a specific topic or area of study. DCA does not have a model capable of simulating conditions after BLACKER is installed on the DDN. BLACKER provides the "basis for reasonable transition alternative to longer range DDN security architectures that support multi-level secure users, and controlled interaction between such users ..." (DCA, 1987:23) But "several suggested ... research and development topics" need further study. (DCA, 1987:24) A model must be developed to address each of these issues. Use of the BLACKER device will effect the DDN system performance characteristics relating to throughput and delay. The BLACKER will result in some network delay on each packet, but should net a smaller overall DDN system time delay due to more available links for transmission.

III. Background Data and Model

Methodology

The objective of this thesis research was to develop a model to determine what impacts BLACKER would have on the existing capabilities of a network of the DDN and to investigate the transition of that network as it integrates into the single classified backbone segment. The segment under investigation consisted of two networks. The thesis addressed the effects of the BLACKER system on the following performances of a segment of the DDN: message and packet throughput, host throughput and input limitations, packet and message delays, and system bottlenecks (either host or node). In order to investigate the characteristics of any network in a pre- and post BLACKER environment, it is necessary to develop a technique to measure these network characteristics. It is also helpful to have an environment in which these parameters are readily available. BLACKER is in the testing (and final development) stage. Historical data is not available to show the effects of the BLACKER implementation on the network or segment, but limited data is available on the BFE equipment and the results of a small baseline testing. Data referencing the networks is also limited and controlled due to the classification of the various networks of the classified DDN segment.

With the limitations listed above, analysis does not lend itself to a straightforward analytical technique. The

system under investigation is also so complex that it can not be represented by a mathematical model that could be analyzed directly. Simulation is a technique to evaluate complex systems whereby experiments are performed on a model in some orderly fashion. These experiments are designed to enhance the understanding of the system under investigation. According to Pritsker, simulation models can be gainfully utilized at four levels:

1. As explanatory devices to define a system or problem;
2. As analysis vehicles to determine critical elements, components and issues;
3. As design assessors to synthesize and evaluate proposed solutions;
4. As predictors to forecast and aid in planning future development. (Pritsker, 1986:1)

Model Construction

A simulation model was developed using the SLAM II simulation language. (Pritsker, 1986) The model constructed for Network 1 simulates a generic DDN network with 17 nodes and 25 links; the model constructed for Network 2 consists of 10 nodes and 14 links. The database constructed for Network 1 was based on an existing network of the DDN. (Damon, 1988) Network 2 primarily consists of a subset of the links and nodes of Network 1, it also contains 4 additional arcs not in the original design. Appendix A contains information referencing the models used in this thesis. See Figures 3 through 5 for a network representations of the models.

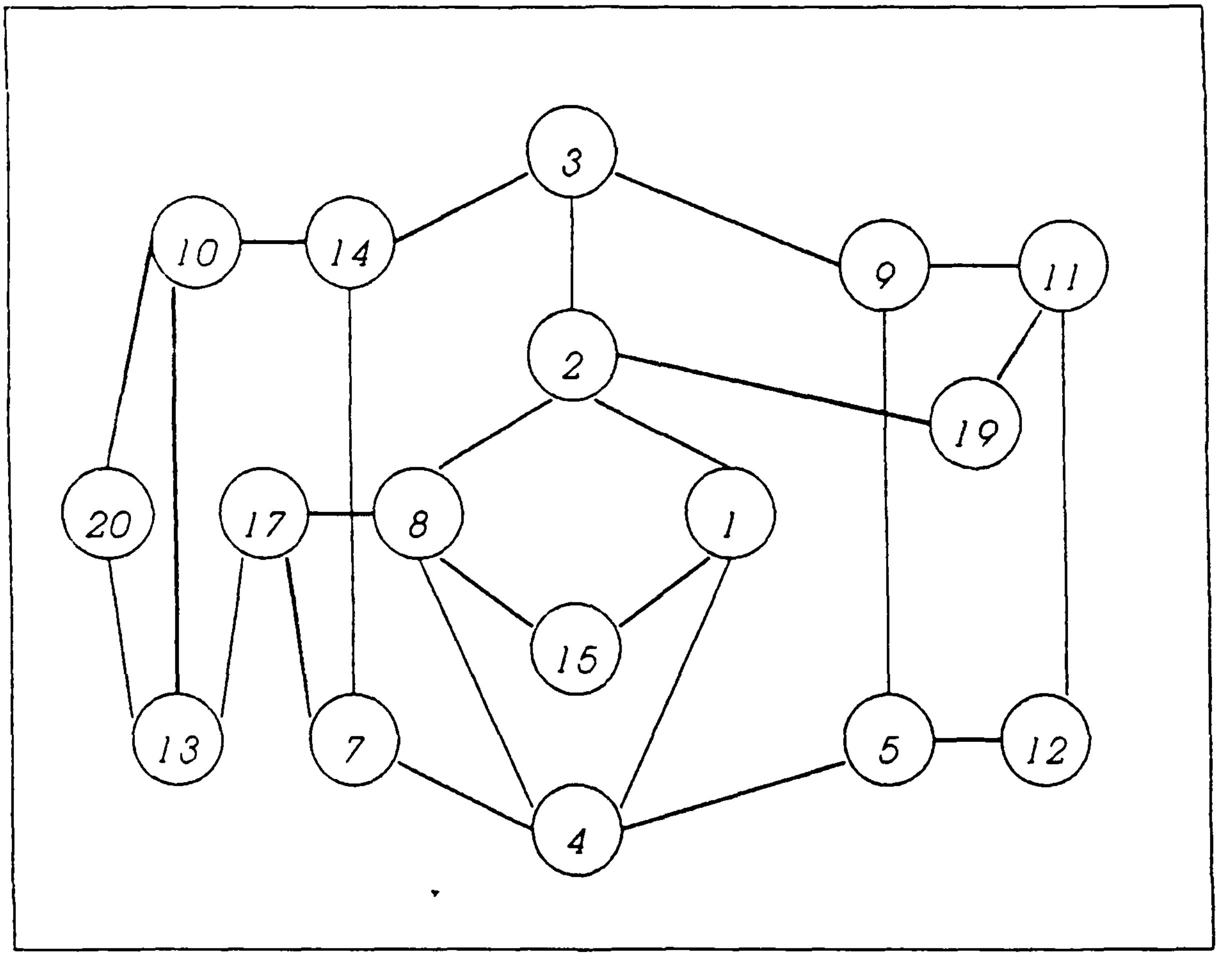


Figure 3. Model Design for Network 1

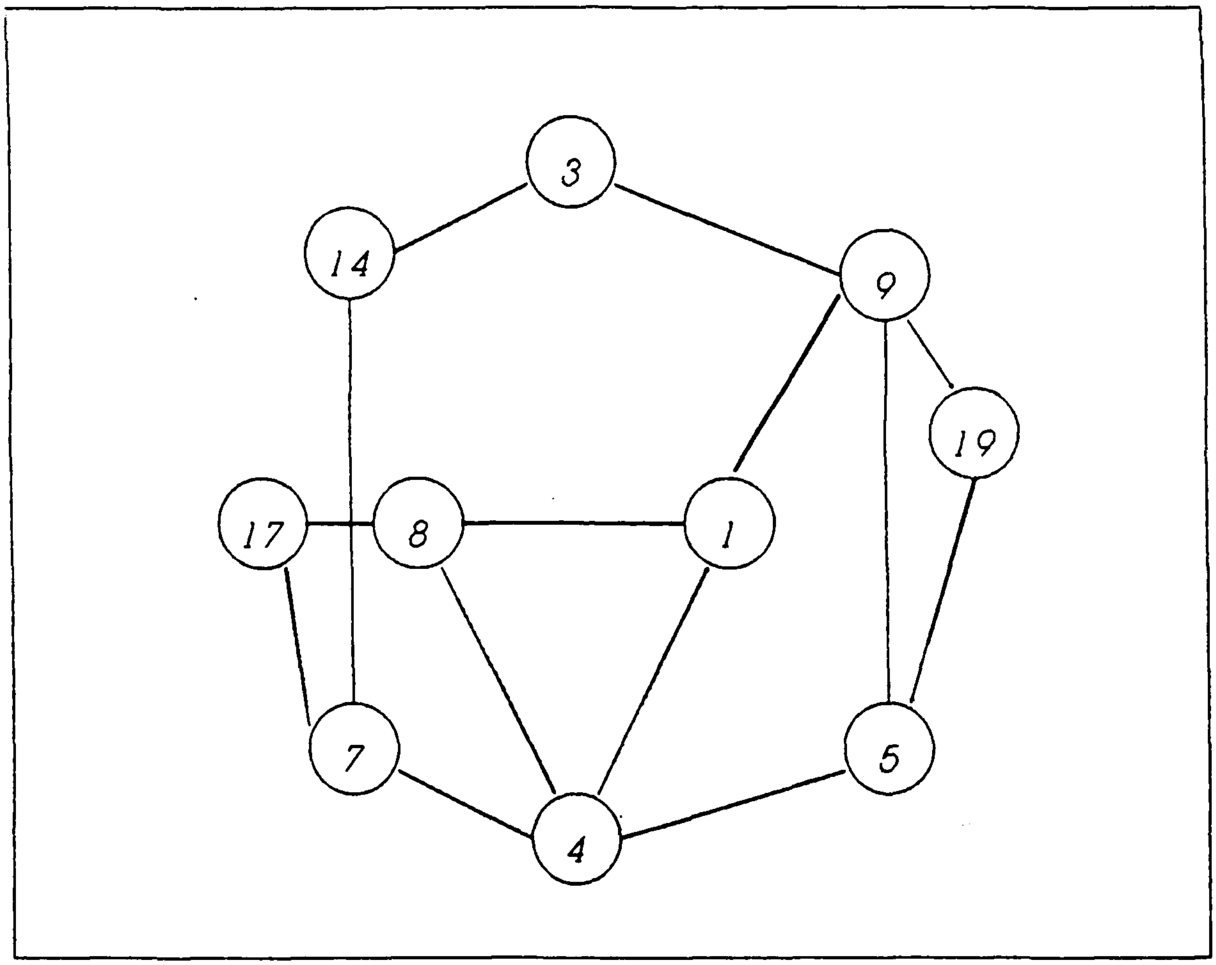


Figure 4. Model Design for Network 2

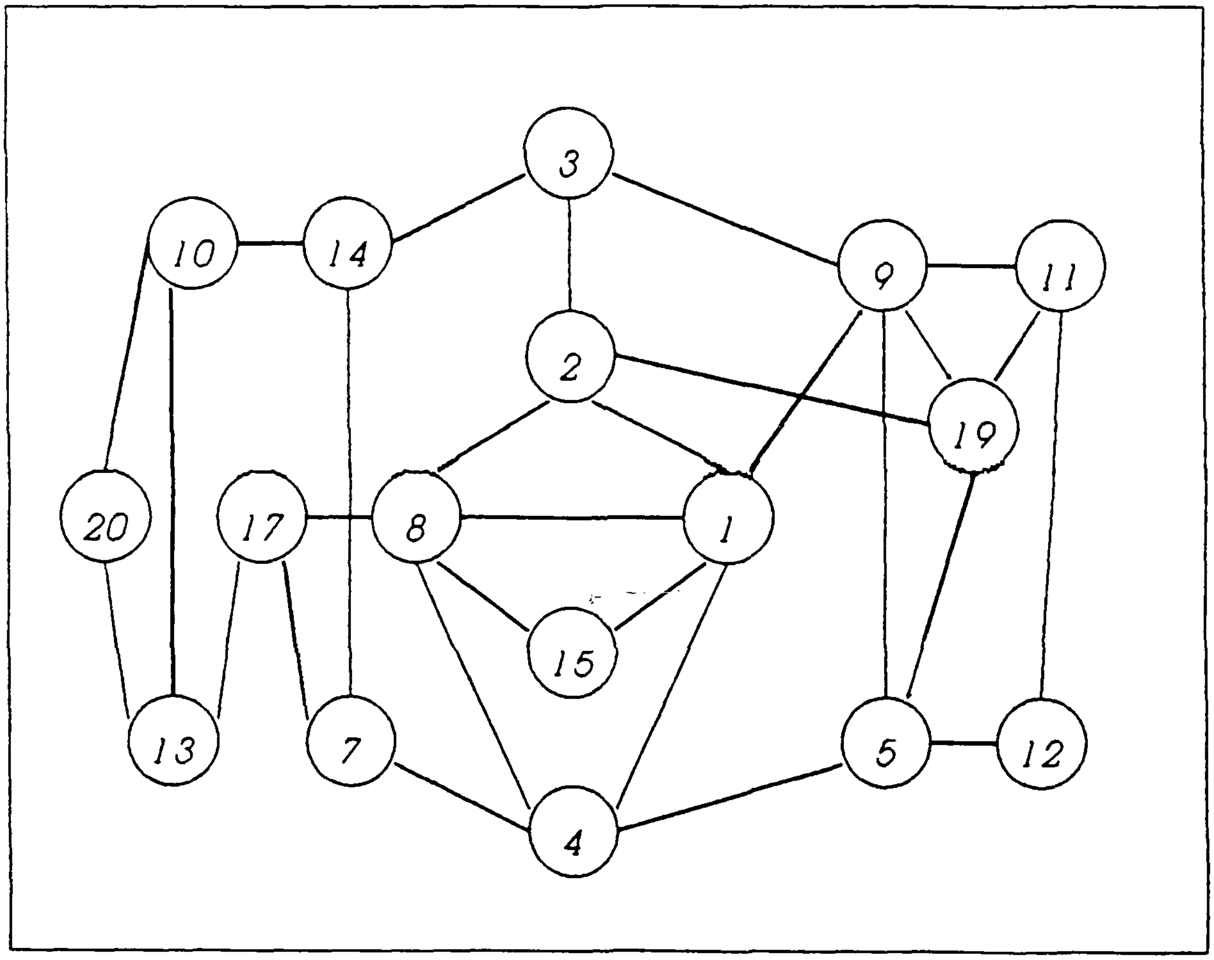


Figure 5. Model Design for the Segment

Packet traffic flows in either direction on the ISTs or network links. Within the model, message traffic is randomly generated at the hosts off each PSN. The host breaks the message into smaller data frames called packets and transmits them to its parent PSN. Each PSN receives traffic from input hosts and processes it routing it toward its final destination. Each node also receives traffic over connect trunks (ISTs) and processes it for delivery. If the data packet is addressed to a host off that PSN it is delivered to the appropriate host; if not, then it is processed and routed to a connected PSN on its continued journey over the network. Each packet of data received must be acknowledged. The PSN must generate an acknowledgement (ACK) for each packet received, and accept and process the ACK for each packet sent. A similar acknowledgement process is executed at the host level. The host must generate an ACK to the parent PSN for packets received, and process ACKs for data packets sent to the parent PSN.

The variable names and attributes in Table 5 were used in the SLAM model. The model processes data packets and system supervisory messages on a priority scheme based upon the value in Attribute 1 of the SLAM II entry or the TYPE of data packet. System supervisory messages are ACKs generated at the host or PSN levels, logical link requests (LLR), logical link request ACKs (LLRA), logical link disconnects (LLD), and logical link disconnect ACKs (LLDA). For the

purpose of this simulation, update routing messages and all BLACKER system messages are treated as supervisory messages. All supervisory messages are complete, they contain only one packet.

Table 5. SLAM II Attributes

Attribute	Variable name	Description
1	TYPE	type of packet
2	PKTS	number of packets in message
3	TIME	time message was created
4	OPSN	originating or parent PSN
5	DPSN	final or destination PSN
6	FPSN	following or next PSN
7	CPKT	number of characters in packet
8	CNTR	global message counter
9	PPSN	previous or last PSN

Messages are generated at the user level, a subdivision below the host. Users send the complete message to their connected host to process for DDN travel. The host sends a message ACK to the user to acknowledge message acceptance. The host must now establish a call setup (or logical link) for the transfer of message data.

The LLR, generated at the originating host, is addressed to the destination host. The LLR is to establish a virtual circuit interface. The destination host returns a LLRA to the originating host with the call setup acceptance and the transmission agreed parameters. After the virtual link is established, then the originating host begins transmitting

the message data packets to its parent PSN. These packets are in a form agreed upon in the call setup. The parent PSN sends ACKs to the host for packets received, and then routes the packets to a connected PSN enroute to its destination. Each PSN in the network continues to route the packets over the backbone network toward its destination based upon the value of the DPSN. Each PSN generates an ACK to the PPSN for the packet when received, and accepts an ACK from the FPSN when its transmits the packet over the connected IST.

When the packet reaches its destination PSN (DPSN) it is then delivered to the destination host. This host groups packets of the same message together and delivers the total message to the end user. The end host must ACK its parent PSN for data packet received. When all packets of the message are received as determined by the PKTS value, then a LLD is transmitted from the destination host to the originating host. Upon receipt of the LLD, the originating host then responds with a LLDA.

When a packet arrives at any PSN, it receives a PPSN value based upon the previous (or original) node and a FPSN value based on the next node as determined by that PSN's look-up table. If the packet has just entered the network and is at the parent PSN, then the PPSN is the same as the OPSN. If the packet has arrived at the destination PSN, then the FPSN is equal to the DPSN. Each node determines the next PSN (or FPSN) that a packet will take enroute to

its destination; this FPSN is obtained from a look-up table developed by the routing algorithm (explained below).

Each PSN is designed as a QUEUE node. Preemption is available but was not a part of the model design since it is currently not a functioning part of the DDN network design. (DCA, 1986) Entries are ranked in the queue based on the value of TYPE. A priority sequence is enforced at the PSN level by processing all trunk traffic first, outgoing host traffic, and then input host traffic. The host processes all PSN inputs before user inputs. Table 6 displays the values used in the TYPE field (Attribute 1).

Table 6. Model Packet Types

Value	Type	Description
30	ACK	PSN to PSN ACKs for BLACKER packets
36	REKEY	KDC to PSN rekey message
3X	BLACKER	BLACKER packets (ACC to KDC, etc)
40	ACK	PSN to PSN ACKs for normal IST packets
46	Update	PSN to PSN routing updates
4X	IST	packets on ISTs between PSNs
50	ACK	ACKs generated to host from parent PSN
51	LLR	LLR to destination host
52	LLRA	LLR ACK to originating host
53	LLD	LLD to message originating host
54	LLDA	LLDA to message destination host
55	Data	data packet to destination host
56	Message	message to host from end user
57	EOM	end of message sequence to host
61	LLR	logical link request to parent PSN
62	LLRA	logical link request ACK to parent PSN
63	LLD	logical link disconnect to parent PSN
64	LLDA	logical link disconnect ACK to OPSN
65	Data	data packet from host to parent PSN

BLACKER System Requirements

The BLACKER system includes four major components. The first component considered is the E3 device or the BFE. The BFE operates as a host front end; to the host, it provides the functions of the PSN; to the PSN, it appears as the original host. When a packet is received from the host, the BFE determined via a look-up address table if the originator and the destination have previously been cleared to communicate. If the destination's address is in the BFE's look-up table, then the packet is forwarded over the network. (Messah, 1987)

These original addresses were stored in the BFE by the second component of the BLACKER, the BLACKER Initialization Carrier or the BIC. The BIC is a hand held cryptovariable carrier used to initialize or update key variables in the BFE. The BFE's network identity was initially loaded by the BIC. (Messah, 1987) Data in the BIC is site unique.

If the destination address was not loaded in the BFE by the BIC, then the BFE must poll the ACC for access control. The ACC, the third component, grants access and queries the fourth component. The KDC then generates the appropriate key variables to the originating and destination BFEs to allow them to communicate. Periodically the KDC will distribute new key variables to all hosts within its COI. There is one ACC and KDC in each COI. They can be collocated with each other, other hosts, or be site unique.

Model Data

The database for Network 1 was constructed from Report R4317 and reflects statistics and traffic conditions from an existing network of the DDN. The input traffic parameters extracted from the report are listed in three tables of subroutine INTLC. Table A holds the percentage distribution for destination hosts off the various PSNs; table C contains the percentage distribution for the number of packets per message from the source; and table D contains the distribution for the number of bits in each packet.

This baseline database was modified for Network 2 where all the table rows were equally divided. For example, there were 10 PSNs in the Network 2. Input host traffic to PSN 1 was equally divided to destinations off all 10 PSNs, each PSN of the network received 10 percent of that PSN's input traffic. A similar technique was applied to Tables C and D of the database. The input message arrival rate for each PSN was held consistent to the value for Network 1.

The database for the combined segment was created by combining values of both Networks 1 and 2. The arrival rate of those PSNs common to both networks was doubled. The values of Tables A, C, and D for Networks 1 and 2 were averaged to obtain the combined segment percentages. The database for each of the network models is displayed in Appendix D.

Packet Transmission Time

A crucial aspect of this research was to model the time required to transmit a message between nodes of the network. Report R4317 provided the transmission times from each PSN to all its nearest neighbors, that value is in milliseconds per data word. The report also displays packets per second and bits per second transmitted on the ISTs. By using Tables C and D and a random number generator to determine the number of packets in a message and the number of bits in each packet, message transmission time was determined in the following manner. Since the report is based on a value of 16-bit data words, the packet length was first determined by dividing the number of bits per packet by 16. This value represented the simulated number of data words in the packet and was then multiplied by the IST propagation rate for data words in milliseconds extracted from the Quarterly Status Report. This value represented the transmission rate from one PSN to the next. It would vary in value based upon the FPSN, since propagation rates are different for each individual trunk. Total DDN delay would be the cumulative processing time for all ISTs that the packet travels plus the cumulative PSN processing time plus processing time at the hosts. If the message contains more than one packet, additional time is consumed at the final host awaiting the remaining packets. Processing time for PSNs and hosts include service and queueing time.

BFE processing introduces a small overhead that results in a data packet delay. Estimated delay is 50 milliseconds per BFE. A greater delay will result when the destination host or end user is not pre-stored in the BFE of the originating host. Then the BFE must poll the ACC who in turn contacts the KDC. If both BFEs are in the same domain, the KDC generates keys for that host pair to communicate. (Messah, 1987) A more complex scheme is required if the hosts are in different security domains. This concept was not included in the model and will not be addressed here.

The time of message creation was marked in Attribute 3. That time was saved in all packets of the same message as they crossed the network backbone. A global message counter, incremented for each new message generated, was used to identify each packet of the same message. This value was stored in variable CNTR of each packet. This message counter value was used in a manner similar to a message identifier in the actual DDN. Packet statistics were taken when each packet reached its destination host. At the destination host, packets of the same message were grouped together (restoring the message) for delivery to the destination user. Since all packets of the original message would have the same unique CNTR, they can be grouped at the destination host based on that value. Message statistics were taken on this single message entity when the complete message was delivered to its final destination.

Routing Algorithm

In order to model a segment of the DDN, an algorithm was developed to determine the shortest path from one packet switching node to any other packet switching node of the same network. The DDN switching nodes employ a routing algorithm that automatically adapts to new topologies in the event of a failure of any network component, once that failure has been detected by any PSN of the network.

The subroutine ROUTES was written to imitate the SPF algorithm used by BBN in the DDN. The algorithm calculates the time delay (or cost) from any point 'A' to point 'B' of a given network and determines the shortest path given the current network design. Update messages are generated every 10 seconds from each PSN to connected PSNs; these messages contains the time for IST travel between each of the connected switches. The subroutine was designed to be re-entrant allowing multiple calls from multiple PSNs. Code for subroutines ROUTES is in Appendix B. The basic steps of the algorithm were summarized in Chapter II. (Schwartz and Stern, 1980:267)

Figure 2 (Chapter II) contains a network flow diagram of a sample network taken from Mandl. This sample network (because of its size) will be used to demonstrate the concept of subroutine ROUTES. The adjacency matrix notation for the example network was presented in Table 3. Note that traffic can not flow in both directions on all paths. This

limitation is not true of the DDN. The network in Figure 2 is only intended to demonstrate the functionality of the subroutines involved in the model.

The network database for Figure 2 was coded into the INTLC subroutine and loaded as the network database for a simulation run. The time delay for each arc is listed in table F of subroutine INTLC (Appendix E). The database and subroutines involved are shown in Appendix E. If no delay is listed in the F(i,j)th element (the value is zero), then there is no arc from i to j. No routing update messages were generated during this evaluation; thus table F remains unchanged during the test scenario. Table G is used as a work table; if no arc exists from i to j then G(i,j) was set to M, where M is some large number (7777 was used in the simulation). Entry G(i,j) will contain the total distance form i to j when the algorithm terminates. Table H was used to store the routing index. When the algorithm terminates, H(i,j) holds the next node in the series or path from i to j. A print routine was developed to display each iteration of the subroutine allowing each step to be monitored and verified. The routine printed tables F, G, and H. The resultant print of the ROUTES subroutine for the network of Figure 2 is listed in Appendix E. The same procedure used on the sample network was used for the DDN network database. The sample network printout is more concise and demonstrated the functionality of the algorithm.

For example, consider the path from 1 to 2 of the network in Figure 2. Nodes 1 and 2 are connected or adjacent, entry G(1,2) displays the total delay to be 1 (all references are made to values of Tables G and H at iteration 6). Entry H(1,2) displays the next node to be 1. If the path starts at 1 and the next node is 1 in order to get to node 2, then it is the last node in the path. Consider the path from 1 to 5. H(1,5) has a value of 3, indicating that the first node after 1 is 3. The journey has been made from node 1 to node 3 at a cost of 4, cost value was noted in table G entry (1,3). H(3,5) has a value of 3, indicating that 3 is the last node prior to node 5. Node 5 can be reached from node 3 at a cost of 5, total travel cost from 1 to 5 is 4+5=9. Note that the value of entry G(1,5) is 9. Any other cost or path can be traced or determined from tables G and H in a similar manner. Note that these values are the same as those obtained in Chapter II when using Dijkstra's shortest path algorithm.

Routing updates are generated between linked PSNs every 10 seconds. When a routing update is received, (OPSN,DPSN) entry of TabLe G receives an update. It will be modified to reflect the current time delay from OPSN to DPSN. ROUTES then calculates the shortest path from 'A' to all other nodes of the network. Table H contains the next node on the journey from i to j, where i is the current PSN and j is the DPSN. Table G contains the total time delay from i to j.

ROUTES was developed to be called when a routing update message is received at any PSN of the network allowing that PSN to have a current and complete picture of the network to avoiding failed or congested areas.

Model Design

Most of the model components have been discussed as separate segments in the preceding sections. The network was modeled as a multi-user system with 17 nodes declared in the network. The 25 links (50 communications channels) were modeled as single server Activities preceded by the PSN QUEUE. Each channel uses a unique activity number to represent a one way communication link. There are two channels in each two way communications link. The PSNs were modeled by a QUEUE Node, a USERF Activity, and an EVENT Node. The hosts were modeled as an EVENT Node, a USERF Activity, and an UNBATCH Node. The end users were modeled as CREATE and ASSIGN Nodes. The ASSIGN Node was used at the user level to define user unique attribute values. The time delay at each processing level was an argument of the USERF. Hosts and end users were generically grouped off the parent PSN with no limitations. With the input parameters given in the Quarterly Status Report, there were no size limitations for hosts. Variations of the baseline network were modeled to answer some 'what if' questions, see analysis in Chapter IV and results in Chapter V.

Basically the simulation proceeds as follows:

1. A message (entry TYPE 56) is created at the end user and passed to the host. The following attributes or variables were loaded at the end user: TYPE, PKTS, OPSN, AND CNTR (see Table 5 for description).
2. The host ACKs the end user for the message and sends a LLR (TYPE 61) to the destination host via the parent PSN.
3. The parent PSN sends an ACK (TYPE 50) to the host and processes the LLR. If the LLR is for a local host it is assigned TYPE 51. If the destination is not local, the TYPE is 41 for IST travel. The PSN assigns the PPSN and FPSN values. The LLR is sent to the appropriate activity based on the FPSN.
4. The LLR is sent over the IST (service activity) based on its FPSN. At the next PSN, an ACK (TYPE 40) is generated back to the PPSN. The PPSN and FPSN values are updated; if this PSN is the DPSN, the TYPE becomes 61 for delivery. If not, it continues in this fashion toward its destination PSN.
5. When the destination host receives the LLR, it ACKs its parent PSN (TYPE 60) and returns a LLRA (TYPE 62) to the LLR originator.
6. This LLRA returns over the network to the originating host in a manner similar to the routing of the LLR.

7. When the LLRA is received at the message originating host, the host ACKs his parent PSN and prepares the message for network travel. The LLRA sets up the virtual link with the two hosts and establishes setup call parameters. An UNBATCH Node makes PKTS copies of the packet entry (TYPE 55). The number assigned to PKTS is based on the data distribution in Table D.

8. These data packets are then routed to the parent PSN for network travel. At the OPSN, the CPKT is determined by a random number based on the database distribution in Table E. Each packet is ACK'ed from the host to PSN, PSN to PSN, and PSN to destination host.

9. When the destination host received PKTS packets with the same CNTR value, the message is complete and can be delivered to the destination end user. Then the destination host sends a LLD (TYPE 63) to the message originating host.

10. The LLD returns over the network to the originating host. Each node on the path ACKs the previous node.

11. The message originating host ACKs the LLD when it is received and sends a LLDA (TYPE 64) to the destination host.

12. The LLDA then crosses the network to the destination host in a manner similar to those above.

13. Routing Messages (TYPE 46) are created every 10 seconds and loaded in each PSN's QUEUE. These messages are routed to all connected PSNs from the originating PSN. When the routing message arrives at the DPSN, the appropriate update in made in Table F, an ACK is returned to the OPSN, and the routing message is dumped. Subroutine ROUTES is called to determine the shortest path for this PSN to all PSNs based on information in the update.

The following steps were added to the model to simulate the effect of the BLACKER.

14. A BFE is added between each host and its parent PSN.
15. The additional BLACKER header is added to each packet when it is processed by the BFE after leaving the originating host.
16. At the destination BFE, the BLACKER header is removed before the data packet is delivered to the destination host.
17. Periodically the KDC will generate rekey messages to each BFE.

Fortran Subroutines

Subroutine INTLC is used to load the network database with the initial traffic distributions and values that are used in determining the routing tables. This subroutine loads the initial seed values used in GENRN to generate random numbers for the data distributions. INTLC calls

ROUTES to determine the initial routing before the first routing update messages are generated.

Subroutine EVENT is used to call other subroutines based on a preset parameter. EVENT is called when an entry arrives at a host, when an entry arrives at a PSN, when the destination host receives PKTS packets of the same message, and when a packet arrives at a BFE. EVENT is also used to generate the rekey messages for the KDC, to generate routing message updates at each PSN, and to generate an output print file when a packet has been misrouted over the network.

Subroutine HOST is used to process an entry that arrive at a network host. It prepares all entries in packet form to be transmitted to a parent PSN and receives all packets from the connected PSN for end users. All entries to the parent PSN from a host will have a TYPE value from 60 to 65. All entries to the host from the parent PSN will have a TYPE value of 50 to 55. TYPE 56 is the message entry received from the end user. TYPE 57 indicates an end of message (EOM) sequence received at a destination PSN after all packets of a message are received. HOST sends the appropriate ACK (type 60) to the parent PSN when data packets are received at the host.

Subroutine SWITCH is used to perform the PSN processing. It received data packets from connected hosts, and routes then over the network toward their destination. SWITCH generates an ACK for all traffic received and processes an

ACK for all traffic sent. Traffic at the PSN is ranked and processed based upon a priority value in TYPE; the entries are ranked low-value-first based on the value of Attribute 1. Packets with type lower than 40 are of BLACKER origin. Packets of TYPE 40 to 47 are from IST inputs or outputs. Packets with TYPE 50 to 55 are destined for a host off this PSN. Packets with TYPE 60 to 65 are inputs to the PSN from a connected host. Subroutine SWITCH determines the next PSN (FPSN) based on a lookup value in Table H of INTLC. Switch calculates the time delay from a PSN to all connected PSNs and updates that value in Table F of INTLC.

Subroutine DTFPSN determines the FPSN based upon the current PSN and the DPSN. The value is obtained from Table G, a lookup table prepared by INTLC and updated by ROUTES.

Subroutine ACK is called by hosts, PSNs, and end users to generate the appropriate acknowledgements for message and packet entries.

Subroutine USERF is used to determine the time delay at the BFE, the host, and the PSN. Values for the BFE were calculated at 64 kilobits per second. (SDC, 1985:Ch 2, 1) Processing at the PSN was calculated based on the following estimated throughput: for PSN to PSN connections, 175 packets per second; for host to PSN connections, 100 packets per second; and for host to PSN to host connections, 160 packets per second. (DeVere and Passmore, 1986:Ch 5, 15a) Host processing time was set to .001 seconds per packet.

Subroutine ROUTES is modeled after Floyd's Algorithm and prepares Table H with the total distance and Table G with the next node in the journey from i to j with the 'shortest route.'

Subroutine RTERS is used to generate the routing update messages from one PSN to another. It was also used to generate the rekey messages for the KDC to the BFEs.

Subroutine GENRN is used to generate a random number based upon a seed value stored in INTLC. Subroutine DTDIST used the random number generated by GENRN to determine a desired output destination, the number of packets in each message, and the number of bits in each packet.

Subroutines REPORT and PRINT were used during model development to aid in design and verification; PRINT is currently used to record any data packets that SWITCH determines to be lost or without a valid path to its DPSN.

Model Assumptions

This model assumes zero processing time at the receiving PSN for supervisory messages. When a packet is received at a PSN, the ACK is generate immediately; there is no time processing delay. A similar procedure was installed at the host level; no delay was realized to generate a LLRA when the LLR was received, no delay to generate the LLD when the last packet of the message is received (EOM sequence), and no time delay to generate the LLDA when the LLD is received.

If the LLR request is not received or is not ACK'ed on the first transmission, the message is dumped at the originating host. The same procedure is preformed in the pre- and post-BLACKER environment.

When all packets of the original message are received at the destination host, then the LLD is generated by that host. In the DDN, either the originating host or the destination host can generate the LLD. In the simulation model, only the destination host generates the LLD.

The time to travel from any end user to its host is unknown and would vary based on the configuration between the two. To model the end to end delay, some value was needed. An arbitrary low estimate of .001 seconds per message was used as the host processing and travel time. This value was realized on both ends, at the origin and the destination.

Supervisory messages were created with a standard length of 10 characters except those listed below. Routing updates were generated with a character length of 21. Rekey messages had a standard length of 20 characters. All data packets ranged in size based on the input distribution from the R4317 Report for Network 1. The character increase for the BFE header was set to 64.

Monitor Centers were not modeled within the simulation. The KDC and ACC were collocated with Packet Switch 1 (PSN1) when the networks were combined.

The BFEs contained all the host addresses in that domain at initial site startup. BFE to ACC and KDC traffic was maintained at a minimum. All traffic generated was addressed to hosts in the designated community or domain. No inter-domain connection was attempted.

Routing Update Messages were generated every 10 seconds from a PSN to all connected PSNs regardless of the current link status or change of status. Nodes on the DDN only send the update if a significant change has been noted during this 10 second period.

Each host receives message traffic from end users according to a Poisson process. The hosts off each PSN were grouped as a single input to the PSN. All end users were grouped as a single end user to this host. Inadvertently, traffic to the single host was modeled using a Poisson distribution with a mean value equal to the actual average input for all hosts off that PSN. Due to the discrete nature of the Poisson distribution, this input created a heavier than normal input traffic pattern. Thus the results of this research are conservative when compared to the actual traffic conditions.

Model Verification and Validation

Model verification was performed as the model was being developed. The SLAM II MONTR/TRACE option was used with the PRINT and REPORT subroutines to verified that each packet and supervisory message was properly crossing the simulated network. REPORT was utilized throughout the development stage in a manner similar to the illustration in Appendix E. As each type of message or packet was introduced into the model, its journey over the network was followed both by the TRACE options and the PRINT subroutine.

Input parameters were varied over selected extreme values to ensure that the model was performing realistically and as expected. Each component of the model was tested and desk checked to insure compliance with the current DDN design specifications and concept. Network 1 was constructed with a database from an existing DDN network. The resulting input statistics and utilizations are close (and in most cases identical) to those of the R4317 Report. (Damon, 1988) The resulting simulated study of message throughput and line/node utilization for Network 1 is considered an accurate reflection of actual performance during the time of the R4317 Report. All resulting measurements are reasonable and correspond to actual data from the Report. The response in the output parameters after BLACKER was introduced is also reasonable and produced results consistent with expert opinion and intuition. (Damon, 1988)

IV. Analysis and Finding

Model Sensitivity

Inadvertently, the Poisson distribution used to generate input traffic created messages at discrete time intervals based on the values designated in the SLAM II code. This distribution created more congestion at the host due to multiple or batch arrivals when comparing to other input distributions (for example the exponentional). Output results from the SLAM II simulation models were thus more conservative when reflecting packet and message delay times than if another input distributions were used.

The model's sensitivity to perturbations in the input variables for Network 1 was examined by varying the input distributions and noting the changes in the model's response. The input traffic parameters under investigation are listed in Table 7. Coded entries of -1/+1 were calculated by varying the highest entry in the row down/up by 10% and scaling the remaining entries in that row an appropriate amount proportional to their current value. Each table contained 17 rows of data, one for each PSN of Network 1. There were 51 total rows of data requiring modifications for each experimental run.

If a two-level factorial design was desired, 2^{51} runs would be required. In a supersaturated simulation model similar to this, Mauro (1986:296-305) recommends the group screening design to identify the most important variables of

the model. This technique was performed in the following manner. A series of simulation runs with all parameters at the coded 0 level, and then all parameters of the first table only at the 0 level displayed only minor variations in the output response. Analysis of the SLAM II Output Reports revealed that the packet and message delays were the only output parameters with any significant change, thus these two outputs were used to gauge model sensitivity. These tests and the resulting assumptions were made based on prior knowledge of the system, a concept Mauro cites as extremely useful in filtering unimportant factors. All entries of table C were grouped as a single factor; also, all entries of table D were grouped as a single factor. Thus, the number of parameters under investigation reduced to 19 (17 rows of data in Table A, and one entry for Tables C and D each). These 19 variables were then introduced into an orthogonal Plackett-Burman (PB) design with N=20. Table 7 listed the variables used in the models.

A stepwise regression was performed on the responses noted above with the results given in Table 8. The table does not list any entry with an F-value greater than 0.10. The PB design used to generate the responses is illustrated in Appendix F.

Table 7. Input Variables used in Regression Models

Variable	Description of Distribution (dist.)
A	table* with dist. for destination hosts
C	table* with dist. for number packets per message
D	table* with dist. number of bits per packet
A1	PSN1's input destination dist.
A2	PSN2's input destination dist.
...	**
A20	PSN20's input destination dist.
C1	PSN1's dist. for number of packets per message
C2	PSN2's dist. for number of packets per message
...	**
C20	PSN20's dist. for number of packets per message
D1	PSN1's dist. for number of bits per packet
D2	PSN2's dist. for number of bits per packet
...	**
D20	PSN20's dist. for number of bits per packet

* table contained 17 entries, one row for each PSN
PSNs are numbered 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 19, and 20

** individual input distributions are grouped in the table designated by the prefix character

Table 8. Summary Stepwise Regression Procedure

STEP	VARIABLE ENTERED	PARTIAL R**2	MODEL R**2	F	PROB>F
1	C	0.5769	0.5769	24.5384	0.0001
2	A15	0.1289	0.7058	7.4492	0.0143
3	A14	0.0633	0.7691	4.3896	0.0524
4	A12	0.0611	0.8302	5.3993	0.0346
5	A7	0.0547	0.8849	6.6467	0.0216
6	A2	0.0410	0.9259	7.1887	0.0189
7	A1	0.0160	0.9418	3.3012	0.0945

See Appendix F for the regression of the full model. The residual plot is scattered, no pattern or transformation is obvious. The resulting model for the i-th response from the regression above was

$$R_i = .66269 + .03319*C + .01710*A15 - .01020*A14 - .00999*A12 - .00939*A7 - .00799*A2 + .00670*A1 \qquad (3)$$

By a modified group screening design, the 7 variables were introduced into the second series of runs to determine the most significant factors. The results for the regression on the 7 parameters above is listed in Appendix F. The resultant model from a step-wise regression for the 7 parameters above was not significantly different from the previous model for all 19 variables,

$$R_i = .66564 + .03214*C + .01011*A15 - .00703*A14 - .00634*A12 - .00411*A7 \qquad (4)$$

The Pearson Correlation Coefficients (output of SAS's proc CORR) between variables of each model were not significant, the highest value was 0.16149. The variable D when varying over a 10% range had no significant effect on the model response. The parameter estimate for D was 0.004358 with a standard error twice that value. An alpha value must be greater than 0.6085 in order to reject that its parameter estimate was not zero. Thus varying entries of D or the entire table over a 10% range has no significant

influence on the model's response. Therefore the variable D was removed from the model.

The A variables in the model came from the input destination distribution; both models indicated that traffic from A15, A14, A12,and A7 played a significant role. The data distribution for A15 indicated that most of its traffic was delivered to distant host. Analysis revealed that the majority of A15's traffic (62%) moved three or more PSNs away. Analysis of A14's traffic revealed that 89% was delivered within two hops, and 95% was delivered within three hops. A hop is defined as traveling on any IST between PSNs en-route to the destination. A similar pattern was noted for A12. Traffic for A12 was almost all delivered within four hops, and 78% was delivered within three hops. Analysis revealed that any A parameter with a positive coefficient in the model had a majority of it traffic delivered to destinations that were three or more hops away. Analysis revealed that the larger the coefficient in the regression model, the further away the destinations were from the input PSN. A parameter with a negative coefficient in the model had the majority of its traffic delivered to destinations that were less than three hops away. The examples examined revealed that the larger the negative coefficient, then the greater the amount of traffic delivered close to the parent PSN.

Variable C, the number of packets in a message, was the most significant of all input parameters. In the second model, the interaction between A15 and C was noticeable but not significant. A modified group screening design (Mauro, 1986:301) was employed to combine all the A variables into one group to study the major effects and interactions of each input parameter type. The main areas of concern were the increase in packet length and the input destination distribution, or variables A and D. These two variables will be modified when BLACKER is introduced to the network, variable C will not be affected by BLACKER and will remain constant.

The main effect and interactions of these three variables (where each table was treated as a single variable) is developed in several regression models of Appendix F. First consider the model where the response

$$R_i = B_0 + B_2*C + B_1*A + B_3*D \qquad (5)$$

The model was used only to estimate the main effects and realized an R-Squared value of 0.9754. Results of the regression in Appendix F revealed that the variable D can be ignored unless alpha is greater than 0.2254. Then the model for the main effects becomes

$$R_i = .660114 + .03271*C - .007542*A \qquad (6)$$

The coefficients of A and C are sufficiently small compared to the value of B_0 that regardless of the values of A and C (established by guidelines within this model's database), they will not realize a significant change in the response. Consider the parameter estimates on the full model in Appendix F with all interactions, again only A and C are significant. No interaction terms will be considered at an alpha value of 0.05, in fact alpha must be greater than 0.12 to show any significant interaction. The full model is listed below.

$$R_i = .66 - .0075*A + .0327*B + .002*C + .003*A*B + .0015*A*C - .0001*B*C - .00125*A*B*C \qquad (7)$$

The full model was examined by varying values of A and C from minimum to maximum. The average number of packets per message is 2.02. If that value is varying 10%, the response was only affected by 0.006. The same is true for A, when using the PSN with the largests input distribution the response was only affected by 0.007. In all models developed, A has a negative coefficient and C has a positive coefficient. These two variables have a tendency to cancel each other out. Even if these values were combined in the same direction, they would have a maximum modification of .013, a value not significant enough to affect the large B_0 value. With the values given in the initial study, all three factor become insignificant when compared to the

B_0 value. If the response is to be estimated under the conditions existing during the R4317 Report, then an appropriate model for the response is an average of the responses. The average response value is 0.6579333. From a statistical viewpoint, either equation 6 or 7 would model the responses; from a practical viewpoint, the average of the responses would be sufficient.

The above analysis was not limited to Network 1. The same series of tests were started on Network 2, but the change noted in system response time was not significant over any varying range of the input parameters. All values were set at their coded low and high, and compared with the response for the original simulation run. Several variations between the extremes also realized no difference in the system response. The initial values loaded in the database for Network 2 represented an even distribution. Varying or even doubling some of the column entries had no effect on system response. Network 2 was a smaller version of Network 1. The maximum number of hops for Network 2 was 4 compared to a maximum of 6 for Network 1. Both network processing times (for PSNs and hosts) and IST travel times were identical. The regression equation for Network 2 reduced in a pattern very similar to Network 1, a practical model of the response would be the average response.

Sensitivity Summary

Given the data distributions of the Quarterly Status Report, the model developed is not sensitive to variations in the input parameters. The model was initially executed with limited overhead message traffic. These limitations were removed and the series of runs were performed again. A slightly higher value was obtained for the average message delay (value was 0.660 instead of 0.657), but the results were identical. The same parameters were the major players when reducing the model. The variations in the response data collected were not significant enough to have an affect on the model, the average message delay for Network 1 was 0.658 with a high and low value of 0.712 and 0.597. Estimating the model over a close region, a very good estimator would be the average value of the responses. Over a small change of 10% in the input distributions, the model is not sensitive and responds without affecting system throughput and packet or message system time.

The model's ability to respond to variations in input distributions is a direct result of the routing algorithm. The algorithm, if functioning properly, would route traffic around congested areas and utilize other network resources attempting to minimize system delay. This system routing modification is observed by noting the shift in IST utilization and trunk entry count when the input-destination relationship is varied.

Model Analysis

Experimentation, attempting to answer the objectives listed in Chapter I, often resulted in numerous simulation runs and presented more "What if" questions than time allowed to answer. Some of the ideas considered and examined are listed below.

Effect of BFE Character Increase. The sensitivity analysis reformed in the preceding sections indicated that the model was not sensitive to a 10% variation in the character length. The BFE will introduce an addition 64 characters to each packet. (DCA, 1985:1-25) The average packet length for Network 1, value taken from the R4317 Report, is 110 characters. This increase is 58% of the original data packet. Is this sufficient to have an impact? This character increase was added to Network 1 in Series 3 runs to determine the impact of the character increase. Results of Series 3 runs were compared to Series 2 runs, the same network without the addition 64 character header introduced by the BFE.

KDC/ACC Location. What impact will the location of the KDC and the ACC have on average system response? If these facilities are to be located at a new PSN not currently in the network, where and how will they be connected? The optimal implementation scheme is in itself a complete study. If the KDC and ACC are to be collocated with an existing PSN, which one? Several simulation run were performed on

Network 1 with the KDC and ACC located at a separate PSN, one new to the network. This new site was PSN6, it was connected to PSNs 7, 8, and 9. In order to prevent the network from using these new ISTs to pass normal traffic, the channel activity times were set to a higher than normal value.

The same series of experiments were performed on Network 1 but the KDC and ACC was collocated at PSN1, a node that already existed in the network. The message delay was 0.02 seconds larger on Series 3 runs. This delay was primarily due to the large time delay introduced on the new or addition ISTs. The channel activity times for Series 3 runs were lowered to a reasonable time for the distance involved, but realized a complete shifting in traffic patterns. The routing algorithm, attempting to minimize total time, used these new trunks to pass message packets. The time delay then for Series 3 runs was not significantly different from Series 2 runs.

Routing Algorithm Effect on Model. A series of simulation runs executed when the routing algorithm was not allowed to update the routing table, consistently realized a larger network message delay time. The original routing table for Series 1 runs was used for the entire simulated time. These message delays are contrasted in the following sections.

Table 9 lists the series of simulation runs performed and a brief description.

Table 9. Network Simulation Series

Series	Network Description
1	Network 1 without routing update capability
2	Network 1 with routing update capability
3	Network 1 in 2 state with BFE 64 Character increase
4	Network 1 with full BLACKER system
5	Network 1 - KDC/ACC located at PSN 1 without BLACKER
6	Network 1 - KDC/ACC located at PSN 1 with BLACKER
7	Network 2 without BLACKER
8	Network 2 with BLACKER
9	Segment - Combined Networks 1 and 2 with BLACKER

Series 2, 3, and 4 have new PSN 6 for KDC/ACC
PSN 6 is attached to PSNs 7, 8, and 9.

Host Throughput and Input Limitations

Host input to the parent PSN was not affected to a significant level by the introduction of the BLACKER system. No PSN QUEUE reached its maximum capacity, no data packets were lost from the system due to blockage. Table 10 contains the maximum length of nodes in each network. The maximum lengths of all PSN QUEUE node are given in Table 11.

Table 10. Nodes with Maximum Length

Network	Without Blacker length (PSN)	With Blacker length (PSN)
1	314 (13)	323 (13)
2	99 (8)	96 (8)
Segment	N/A	207 (4)

PSN average utilization for Network 1 increased slightly, a network increase of 0.002 (.374 to .376). PSN average utilization for Network 2 did not change, average stayed consistent at 0.256. There was some shifting of resources both in PSN and IST utilization for Network 2, also true for Network 1. PSN average utilization for the Segment was 0.307, almost midway between the average for the individual networks. This shifting can be explained by the dynamic routing algorithm, when a node or trunk becomes busy or congested then traffic is routed to avoid that node until the average time delay is reduced.

Table 11. PSN Maximum Length

Node \	Network 1 Pre	Network 1 Post	Network 2 Pre	Network 2 Post	Segment Post
1	20	16	13	12	25
2	157	139	*	*	101
3	23	19	15	17	31
4	110	92	65	74	207
5	12	11	14	15	22
7	34	34	42	38	55
8	84	88	96	99	162
9	13	13	24	17	27
10	48	54	*	*	53
11	11	10	*	*	9
12	10	9	*	*	9
13	314	323	*	*	169
14	57	66	45	46	98
15	48	41	*	*	68
17	5	23	31	33	34
19	9	11	18	13	17
20	12	12	*	*	22

* node is not an active part of Network 2
Pre designates Network in the pre-BLACKER state
Post designates Network in the post-BLACKER state

Note the shifting in the maximum length of PSNs 2 and 13. In Network 1, PSN 13 had an average utilization of 0.712. In the Segment, PSN 13 has an average utilization 0.425. PSN 2 dropped from 0.608 to 0.417. The line and node utilization per PSN and IST are grouped closer together in the combined Segment. There are no extremes in the Segment as observed in the individual networks. The backbone appears to round off the peaks and averages system utilization.

Message and Packet Throughput

There was no significant difference in the packet and message throughput statistics for Network 1 or 2 when comparing the pre- and post-BLACKER states. Table 12 lists the message throughputs for Network 1 under various conditions at selected simulated times. The last column listed the number of messages that entered the network from all hosts. This value is the maximum number of messages that could have crossed the network.

Table 12. Network 1 Message Throughput

Simulated Time	Series of Network 1					Input Totals
	1	2	4	5	6	
300	6399	6390	6388	6388	6390	6395
600	12558	12560	12559	12560	12560	12562
900	18763	18769	18774	18774	18774	18779
1200	24935	24927	24934	24932	24930	24955
1500	31223	31223	31217	31222	31220	31235
1800	37744	37746	37742	37741	37732	37759
2100	43918	43919	43919	43921	43918	43922
2400	50206	50207	50208	50207	50205	50218
2700	56409	56410	56407	56407	56408	56423
3000	62469	62472	62472	62466	62474	62488
3600	75064	75071	75072	75069	75069	75080

The packet throughput statistics for Network 1 are very close in comparison to the message stats. Throughput message statistics for Network 2 are almost identical in the pre- and post-BLACKER states, they only differ at most by a value of one.

Packet and Message Delays

A listing of message delays for the various series is provided in Table 13. The routing algorithm was not used in Series 1, but it was used in all other experimental runs. The system delay for Series 1 compared to 2 is longer but not as significant as the IST and PSN utilizations. If a node was part of the original SPF design in Series 1, it was heavy utilized. If is was not in the original design, it had a PSN and connected ISTs utilization of zero.

Table 13. Network Message Delays for Various Simulated Times

	Network/Series								
	1	1	1	1	1	1	2	2	Both
Time	1	2	3	4	5	6	7	8	9
10	.438	.438	.421	.428	.428	.412	.241	.261	.375
20	.525	.534	.592	.569	.554	.522	.226	.238	.414
30	.547	.551	.584	.584	.530	.520	.299	.292	.399
40	.634	.625	.663	.664	.617	.616	.290	.302	.389
60	.609	.597	.618	.641	.624	.606	.291	.312	.409
90	.629	.611	.619	.648	.633	.622	.276	.295	.483
100	.640	.617	.634	.703	.632	.626	.273	.293	.494
120	.674	.648	.664	.717	.653	.669	.270	.290	.519
200	.712	.714	.714	.814	.675	.730	.269	.293	.515
300	.714	.747	.722	.832	.689	.722	.278	.307	.487
400	.697	.726	.714	.798	.696	.719	.302	.316	.472
500	.663	.687	.689	.746	.667	.674	.302	.315	.466
600	.664	.672	.675	.728	.664	.670	.303	314	.464
900	.657	.650	.660	.695	.658	.655	.302	.316	.459
1200	.659	.637	.655	.681	.644	.642	.303	.318	.459
1500	.680	.661	.672	.690	.671	.663	.302	.317	.458
1800	.721	.670	.707	.699	.689	.673	.299	.314	.456
2100	.707	.660	.698	.691	.685	.669	.298	.314	.453
2400	.699	.653	.689	.683	.683	.661	.297	.314	.457
2700	.692	.651	.684	.677	.681	.656	.295	.312	.456
3000	.673	.638	.667	.661	.667	.641	.294	.312	.459
3300	.672	.638	.669	.660	.665	.640	.294	.312	.460
3600	.671	.638	.670	.660	.663	.639	.294	.311	.462

In the remaining series, the IST utilization was distributed over the entire network. The difference in response between Series 1 and 2 is illustrated in the graph below, Figure 6. Notice that after 3000 seconds, the difference in delay averages a little more than .03 seconds. Thus, Series 1 has no significant difference in the message throughput (see Table 12) or message delay. Analysis reveals that Series 2 has the work load spread more evenly over the network to avoid congestion or bottlenecks.

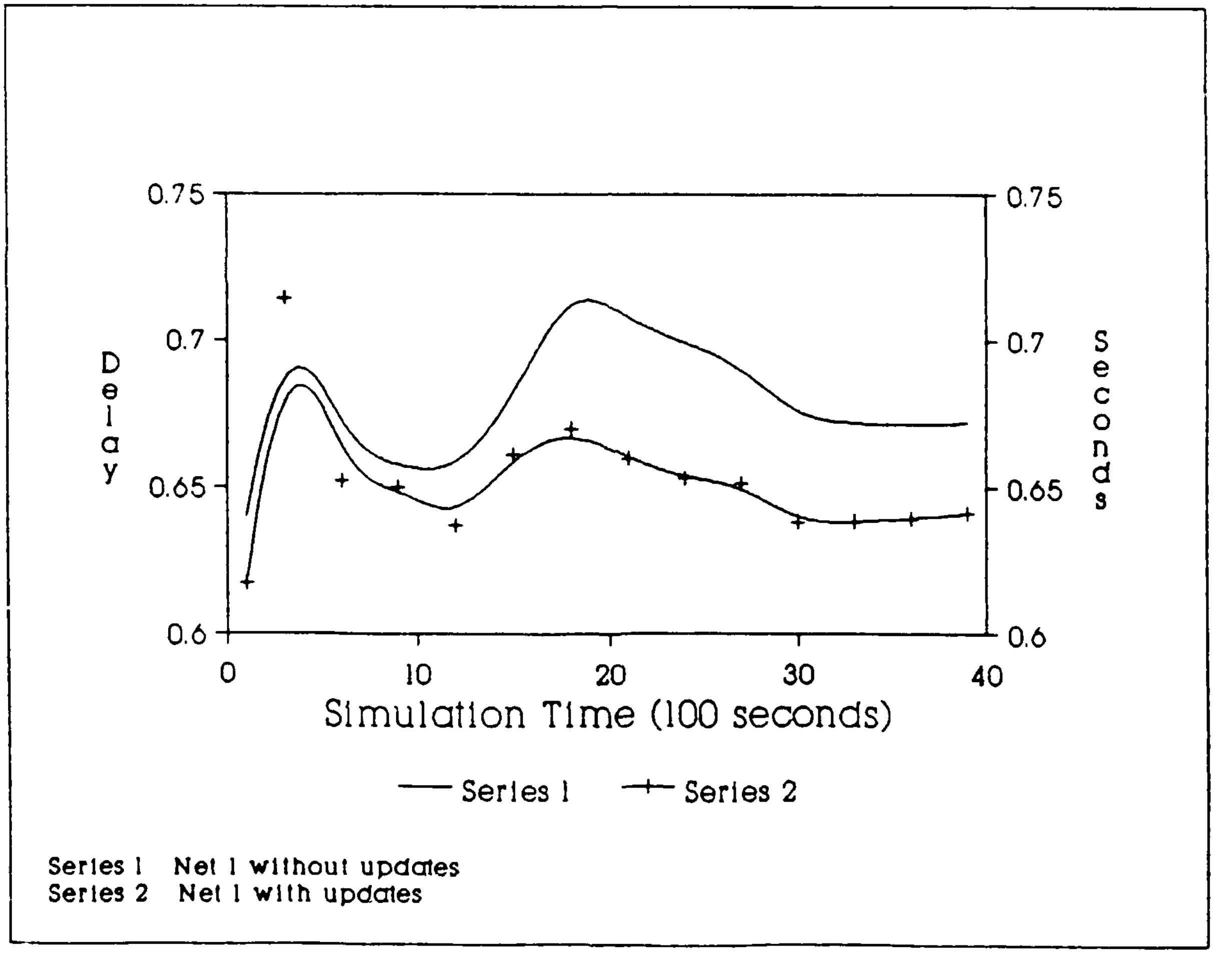


Figure 6. Effect of Routing Algorithm on Network 1

The BFE character increase was simulated in the model of Network 1 or Series 2 and results recorded as Series 3. Then the KDC and ACC were added to the simulation models and results recorded as Series 4. The BLACKER system added a small, but not significant increase to the message delay, and most of this delay can be attributed to the BFE character increase. See Figure 7. The KDC and ACC were attached to the PSNs with the largest traffic inputs. Average IST and PSN utilization was not affected as a result of the BLACKER.

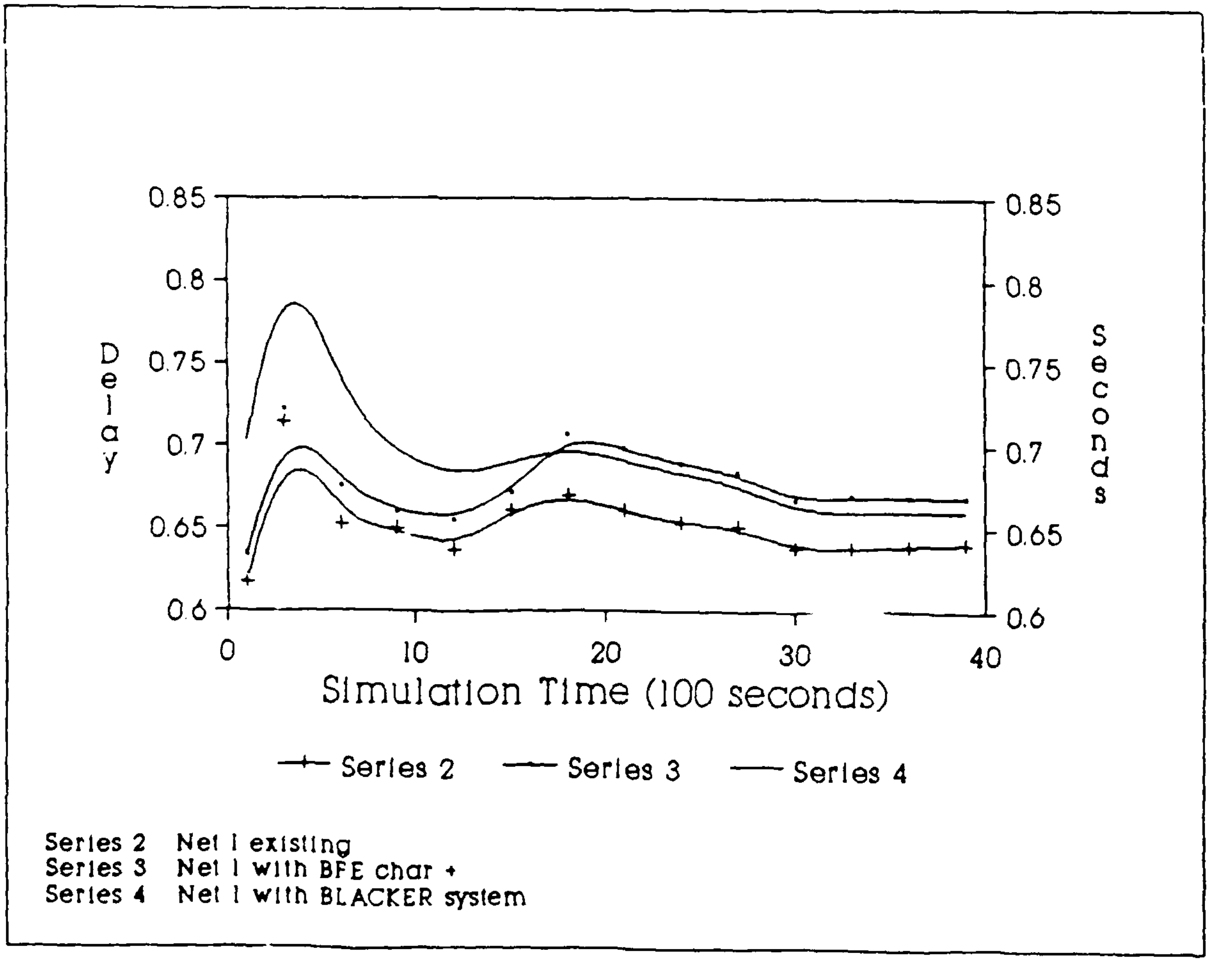


Figure 7. Message Delay in Network 1

The comparison with and without the additional PSN for the KDC and ACC are present below. THe travel times on the connected ISTs to the KDC/ACC were set to a higher than normal rate to prevent the routing algorithm from using them to pass normal data packets.

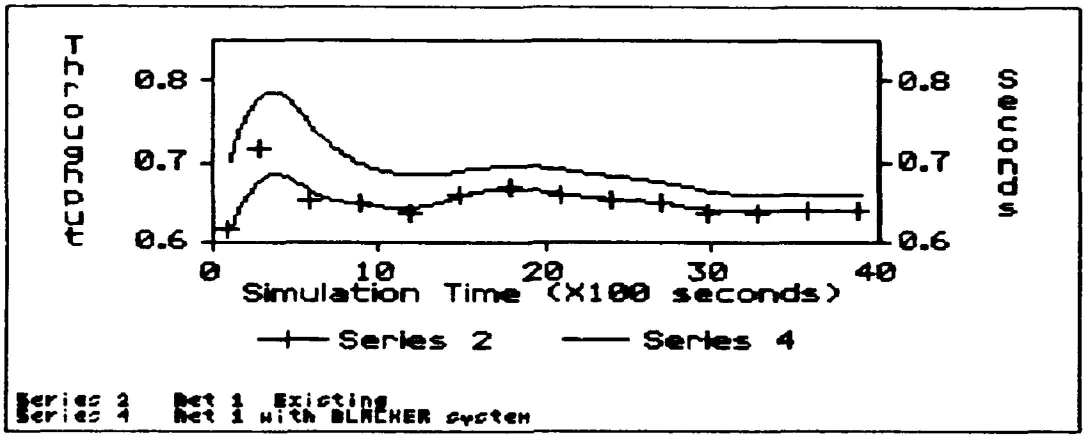


Figure 8. Delay in Network 1 with KDC/ACC Node

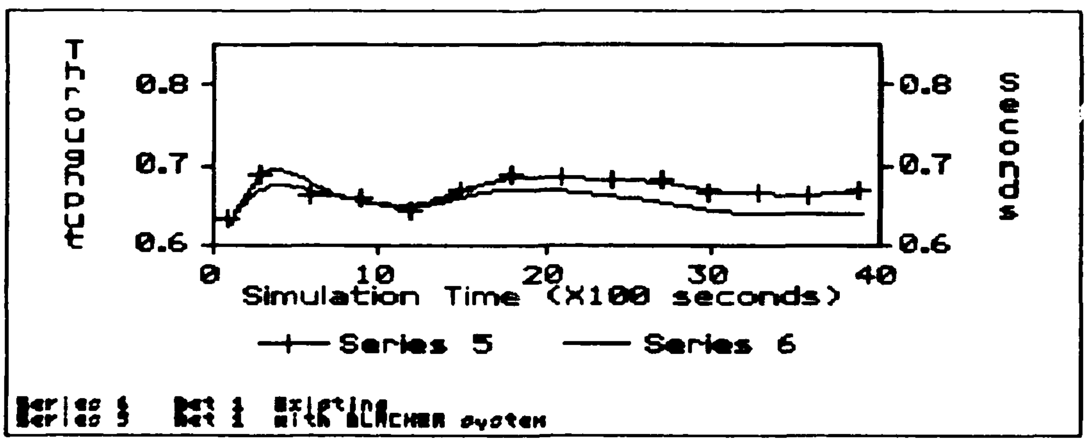


Figure 9. Delay in Network 1 with PSN 1 as KDC/ACC

Message delays in Figures 8 and 9 reveals no significant differences between the two network designs. The same effect was demonstrated for message throughput in Table 13. When the ISTs were given an appropriate propagation rate, the message delay decreased due to more routes available.

Message delay for Network 2 in the pre- and post BLACKER states are graphed in Figure 10. Note that Network 2 has a significant smaller delay than Network 1. Network 1 has an average delay of .638 seconds before BLACKER and .663 after BLACKER. Network 2 has an average delay of .294 seconds before and .311 after.

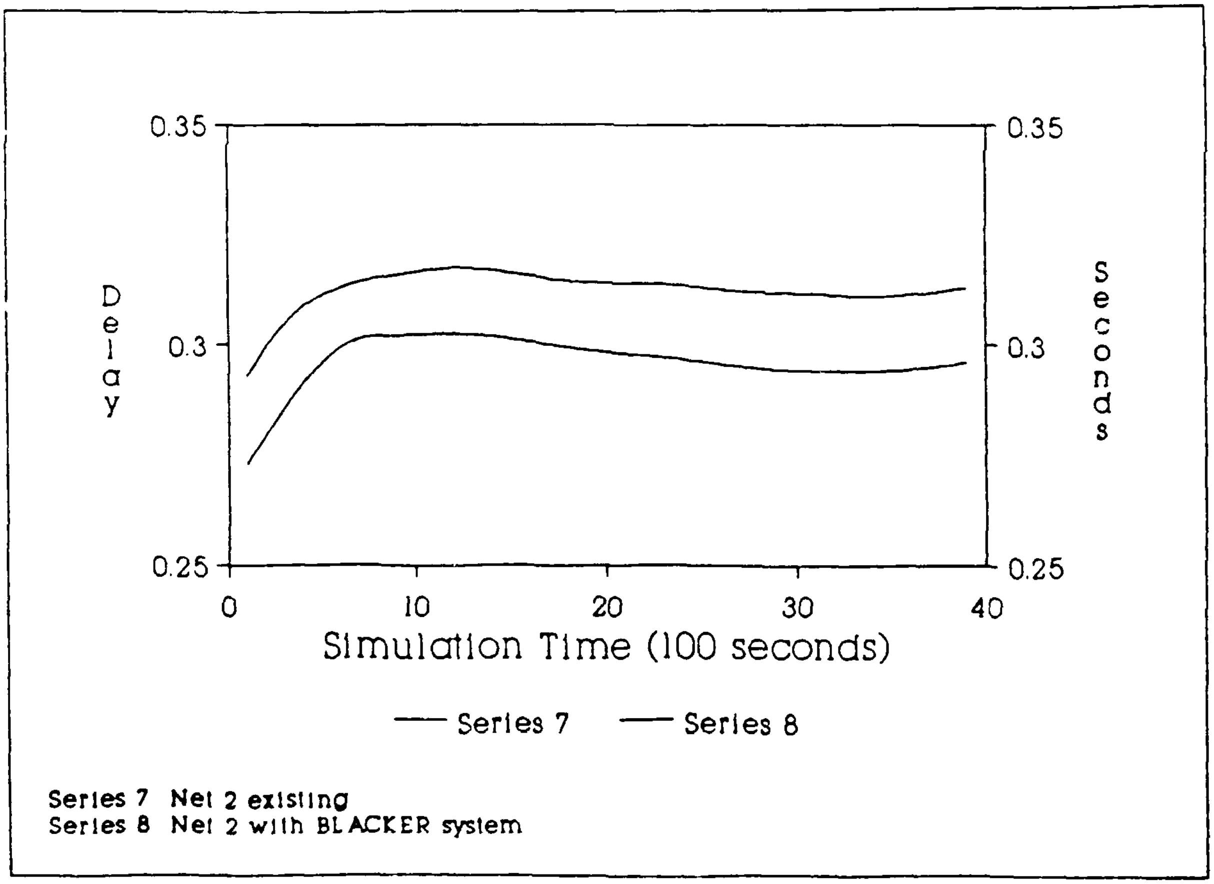


Figure 10. Message Delay in Network 2

Figure 11 displays the simulated message delay on the Segment and compares it with the delay of each network prior to BLACKER. The Segment with Blacker appears to have an averaging effect on message delay. But considering the total number of messages introduced by each network of the Segment, it becomes obvious that the average message delay has been significantly reduced. See the network totals in Table 14.

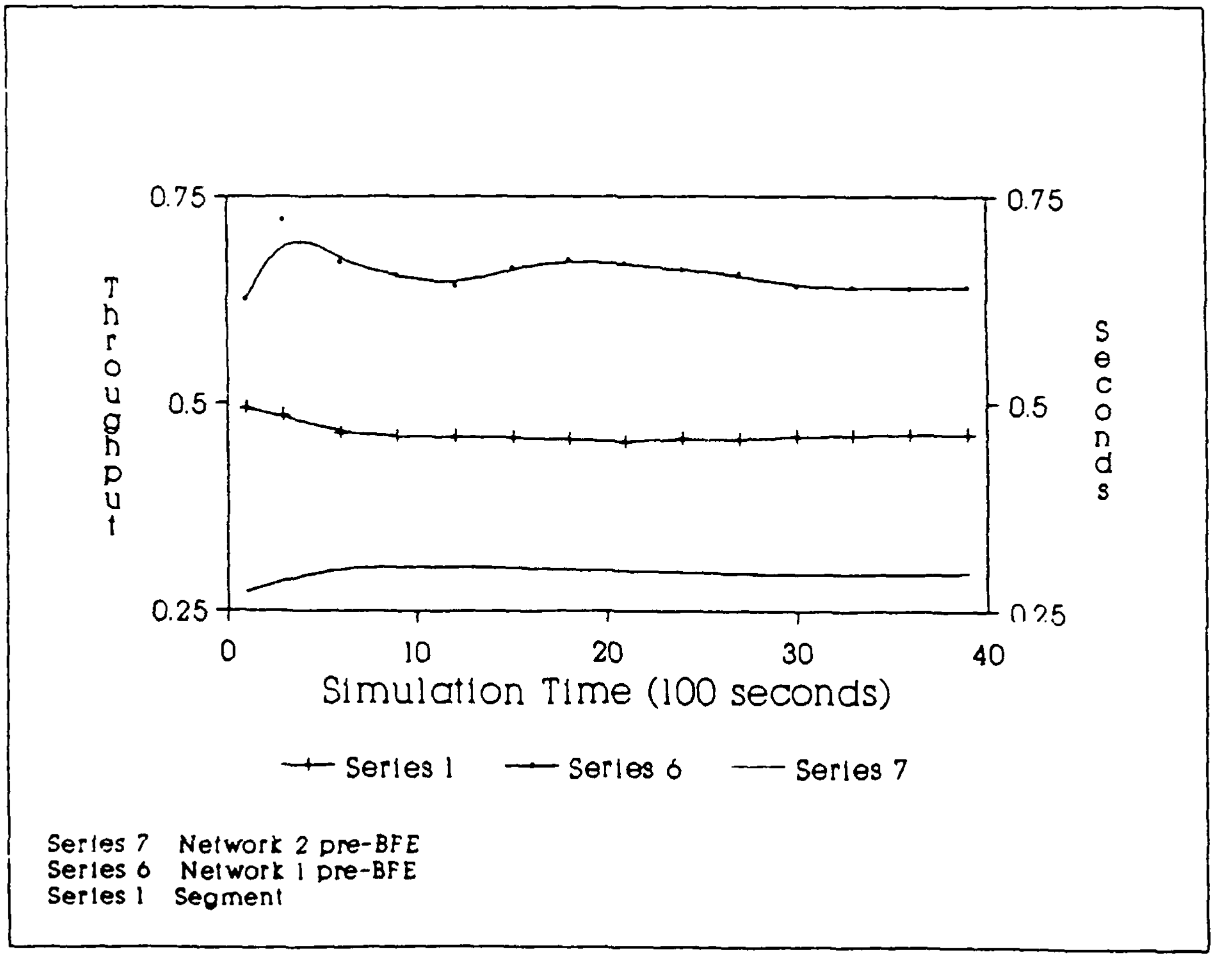


Figure 11. Network vs Segment Delay

Table 14. Message Input per Network

Time	Network 1	Network 2	Segment
2100	43922	18975	61585
2400	50218	21695	70626
2700	56423	24326	79303
3000	62488	26990	88290
3600	75081	32430	106016

* values to illustrate input percentage distribution

Statistics in Table 14 reveals that Network 1 introduces 58% of the Segment traffic when compared to Network 2, and Network 2 introduces 42%. Average message delay for Network 1 is .638 seconds before BLACKER, and .665 after. Network 2 has an average delay of .294 before BLACKER and .311 after. Using the individual network inputs as a weighting factor, the combined average before BLACKER is .493 and .516 after. The simulated average message delay for Series 9 was .465, a value significantly smaller than the combined networks after BLACKER was introduced. Series 9 also removed from the Segment any PSN or IST that was duplicated in the individual networks. Network 1 contained 17 nodes and 25 links, Network 2 contained 10 nodes and 14 links. When the networks were merged, 10 nodes and 10 links were freed. Some of these links could be redirected to provide more routes within the Segment and would result in a smaller time delay.

Figure 12 displays the message time delays on the individual networks in a pre- and post-BLACKER state with the Segment delay. This graph summarizes the contents of the previous 5 graphs.

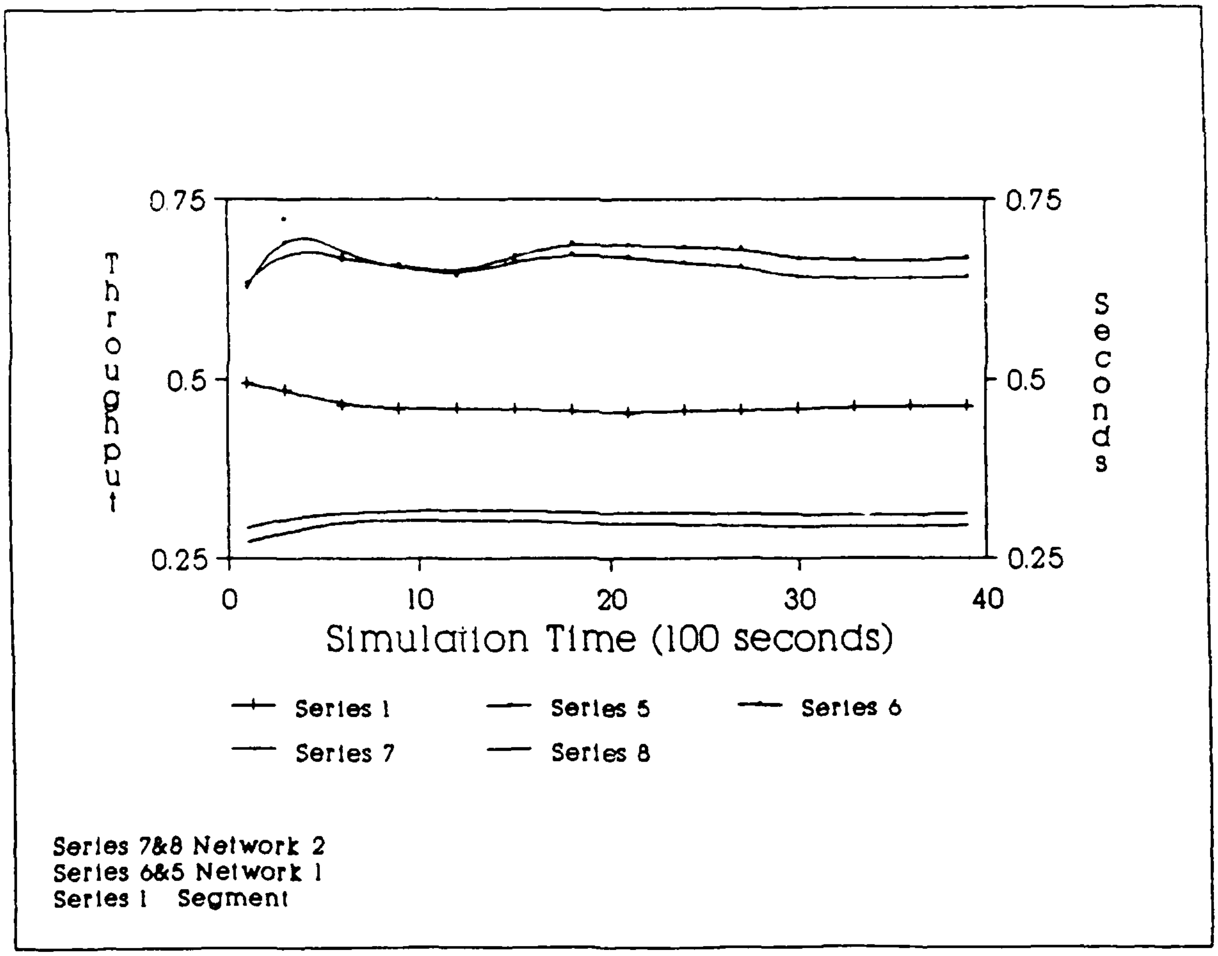


Figure 12. Summary of Message Delays

System Bottlenecks

The traffic conditions introduced into Networks 1 and 2 were not sufficient to create any congested areas. The only PSN bottlenecks and trunks with heavy utilization were noted in Series 1 runs. The PSNs and ISTs designated as the part of the best route in Series 1 runs received a higher percentage utilization than Series 2 runs, but in most of the runs the difference in performance was not sufficient to create a blockage condition.

Traffic was doubled in several runs to observe the resulting effect. Again no congestion or bottlenecks were noted. The surplus of traffic was routed around busy nodes and the network gave the appearance of averaging PSN and IST utilization. There was no effect noted when doubling the traffic in Network 2. Utilization increased slightly on both the PSNs and ISTs, but no delay was realized.

A series of experiments were performed on the Segment without the additional ISTs gained in the merger. Again, no PSN reached its maximum capacity where traffic was being blocked. The average message delay time realized a significant increase to 2.77 seconds. Utilization on both PSNs and ISTs increased significantly, but no bottlenecks were noted.

V. Conclusions and Recommendations

Summary

It is apparent that BLACKER has an impact on the message delay of a network. The adverse effect of BLACKER on the individual network alone can be off-balanced by the additional capabilities provided when the networks are merged in the Segment. This impact on message delay is insignificant from a practical standpoint; Network 1 realized an increase in average delay of .03 seconds after BLACKER installation was complete; Network 2 realized an increase in average delay of .02 seconds after BLACKER. When the networks were combined in the Segment, the system average message delay was .462. Traffic for Network 2 encountered a slowing effect due to the addition traffic of Network 1 on the trunks. But traffic for Network 1 now has more possible routes, and realized a significant decrease in message delay (time decreased from .663 for Series 5 to .462 for the Segment).

Merging two or more networks results in an averaging effect of individual network characteristics. Trunk and node activity and utilization averages themselves between the two networks. When the status and time delays of all the links and nodes are available, the routing algorithm at each node will update and designate the current path with the shortest time (SPF). Each network included in the Segment will provide some additional links and nodes not

previously in the Segment. Thus, each addition to the unified Segment will result in more averaging of network components.

If both networks are identical, then bottlenecks and congestion could result. This effect of combining similar networks that have the same components is identical to increasing the node inputs without providing additional avenues for relief. The switching nodes could eventually reach a peak level when they can not handle the capacity.

Message delay was the only output parameter that demonstrated any significant difference in the pre- and post-BLACKER network performance. With the input data parameters given, BLACKER had no effect on host throughput capability or input limitations. BLACKER created a slight system or network overhead, but it was not significant enough to effect PSN performance or utilization. There was no impact on packet or message throughput in the pre- and post-BLACKER models. With the input database distributions, there were no bottlenecks or blocked nodes in any model examined.

The simulation model provides a means of answers a series of 'What if' questions. The model was not intended to provide the best direction or to predict the only answer. This model was designed to lay a foundation for network modeling and provide a means to study conditions during the merger. Several examples of 'what if' questions are

provided and examined in the research; a brief summary is provided in the Model Implications below. Results of the model are only as good as the parameter estimates used. The methodology used in this research is a reasonable approach to analyzing the effects of merging two or more networks together using the guidelines provided for a secure environment.

Model Implications

The routing algorithm had the following effects on network performance: it provided a more even utilization of network components; it allowed the network to dynamically redirect traffic to avoid congested links; and it provided a smaller packet and message average delay. These results were obtained by comparing outputs of Series 1 and 2 runs.

The effect of the BFE header increase on network performance was noted and compared to the performance with the entire BLACKER system. The increase was significant enough to effect message delay and accounted for the majority of the impact caused by BLACKER. Several rums performed with the addition BFE and BLACKER overhead but without the BFE header increase had no significant impact on message delay or any other performance parameter when compared with the network in the pre-BLACKER model. Analysis was obtained by comparing Series 2, 3, and 4 runs.

The location of the KDC and ACC had no significant impact on the performance of Network 1. A series of runs with the

KDC and ACC both collocated at a new node (PSN6) was compared to the same network with the KDC and ACC collocated with a host off an existing node (PSN1). Link and PSN utilization varies slightly, but differences in network performance was not significant. Results were obtained by comparing Series 4 and 6 outputs.

Recommendations for Future Research

The need to merge classified networks into a unified segment is reality. BLACKER is apparently the current means available to accomplish this, and many questions will appear as time progresses. Some of the future questions envisioned in the course of this research are listed below.

First, which PSN is the best choice for the Key Distribution Center and the Access Control Center? Should they be located at the same site? Should they even be located with another PSN or should a new node be introduced? Second, if a new link is available, where could it be placed to enhance the existing system? If some of the links duplicated in each network can be redirected, what would be the optimal method for allocation? Third, what are the impacts of interdomain messages? What happens when the destination address is not stored in the BFE? Fourth, this thesis limited the architecture to a segment with two networks; what are the impacts of the third, the fourth, ... etc? Fifth, what are the impacts of the next PSN software release on the model performance?

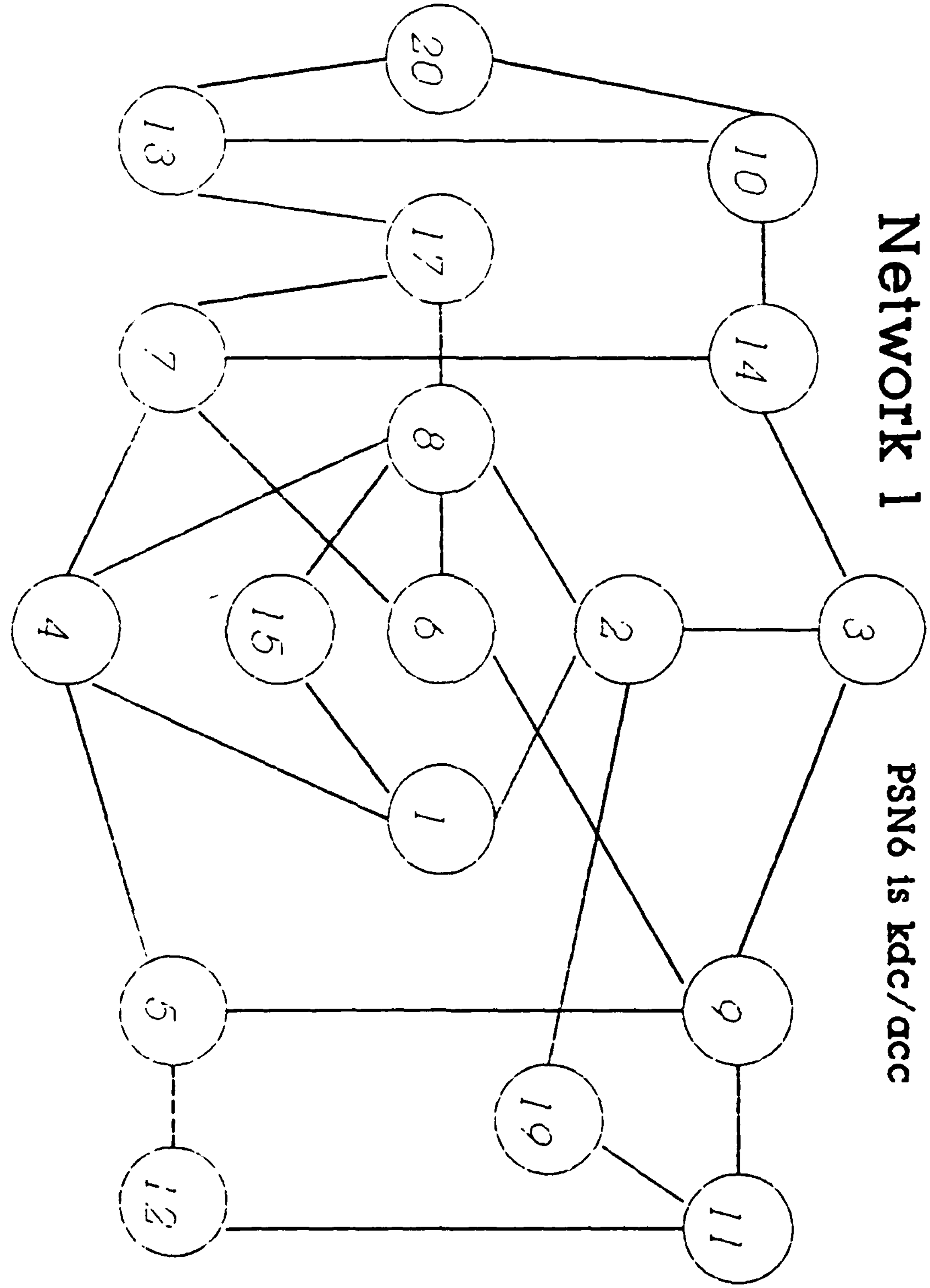
Appendix A. Model Network Descriptions

Listing of Simulation Series

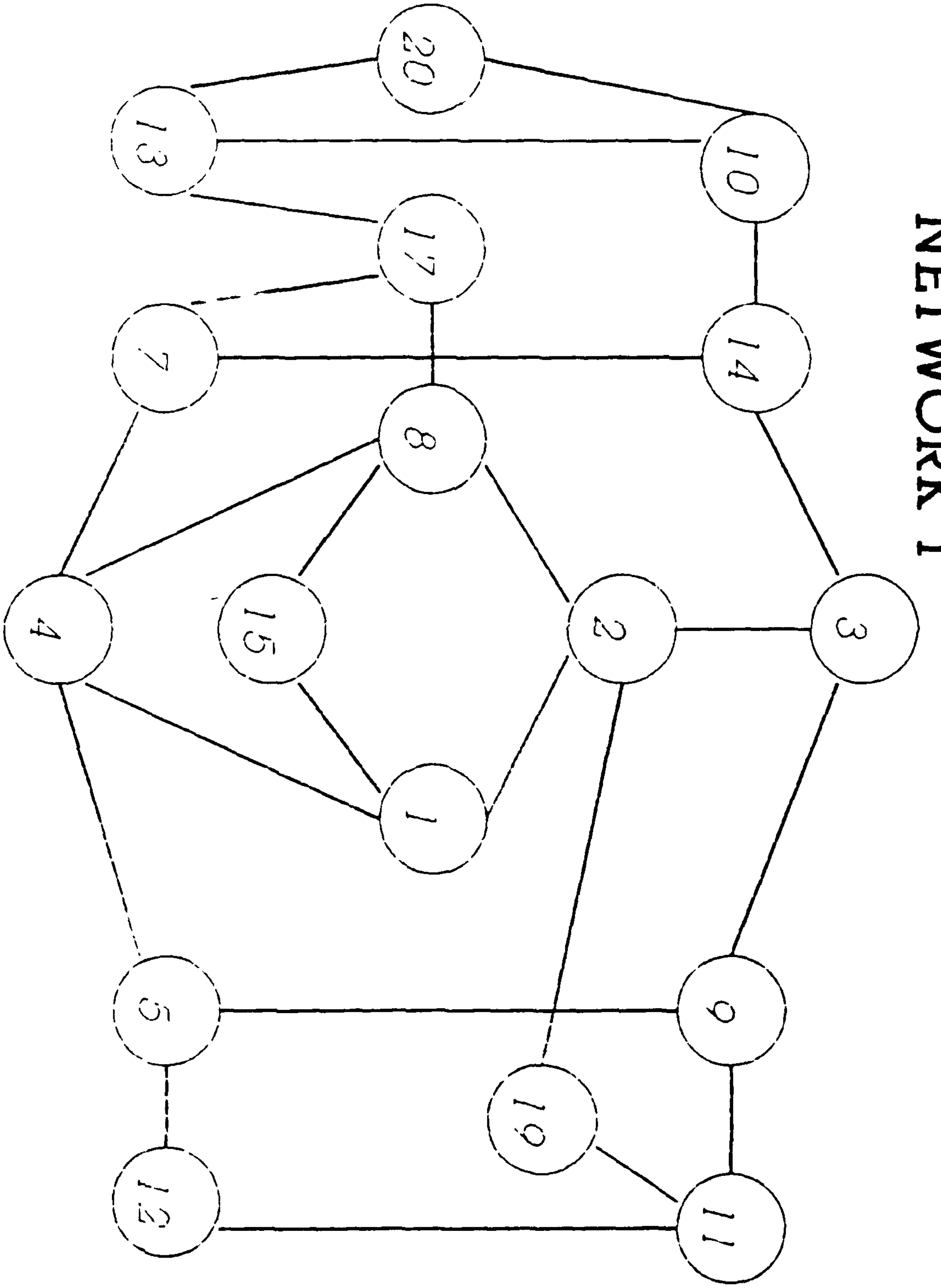
Series	Network Description
1	Network 1 without routing update capability existing network without any BLACKER
2	Network 1 with routing update capability existing network without any BLACKER
3	Network 1 with routing update capability and with BFE 64 Character increase
4	Network 1 with full BLACKER system KDC and ACC are located at PSN6
5	Network 1 - KDC/ACC located at PSN1 without BLACKER
6	Network 1 - KDC/ACC located at PSN1 with complete BLACKER system
7	Network 2 without BLACKER no inputs for KDC and ACC
8	Network 2 with BLACKER KDC and ACC are colocated at PSN1
9	Segment - Combined Networks 1 and 2 with BLACKER KDC and ACC are colated at PSN1

Series 2, 3, and 4 have new PSN 6 for KDC/ACC
PSN 6 is attached via ISTs to PSNs 7, 8, and 9.

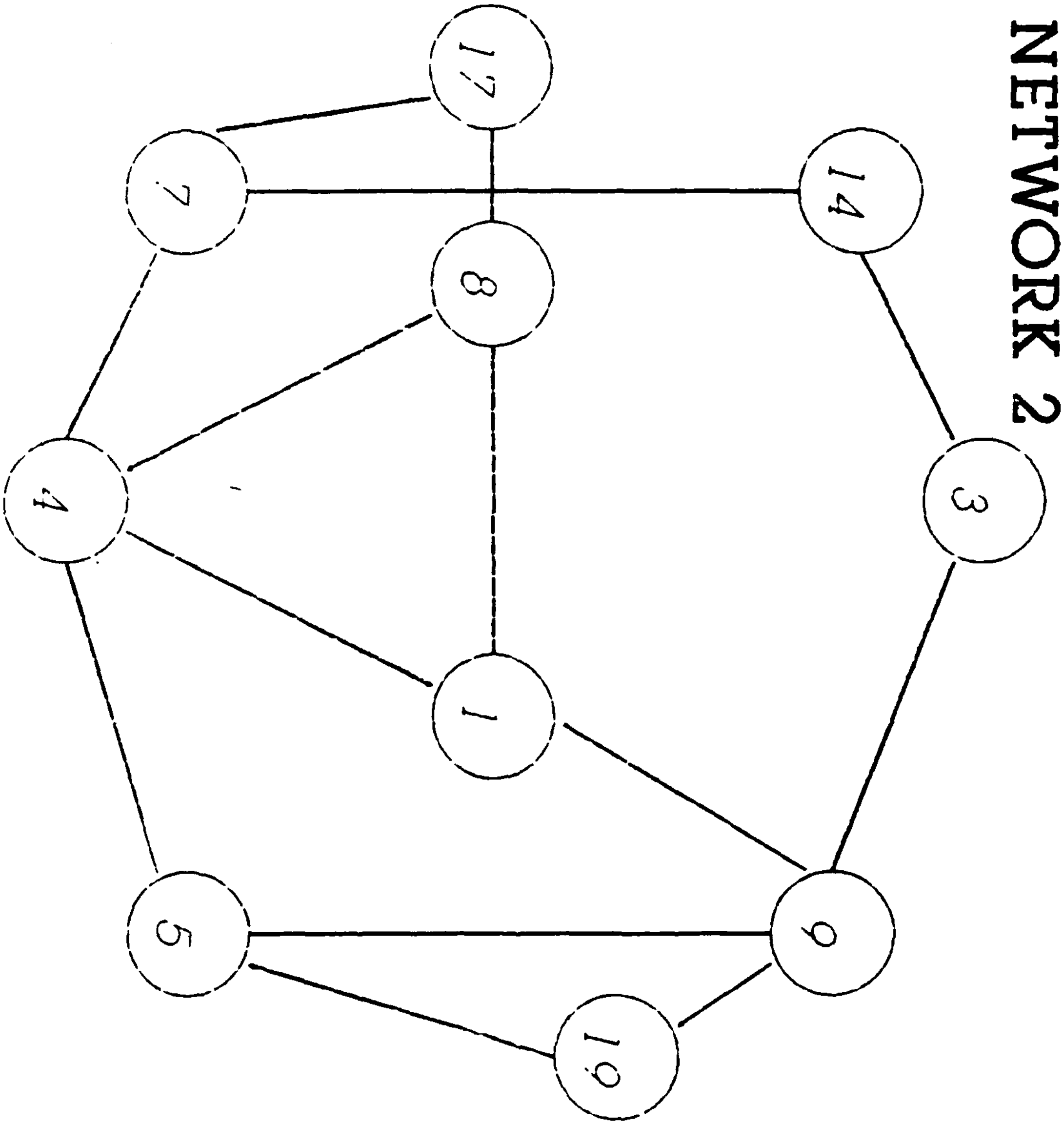
Network 1
ACC/KDC is PSN6



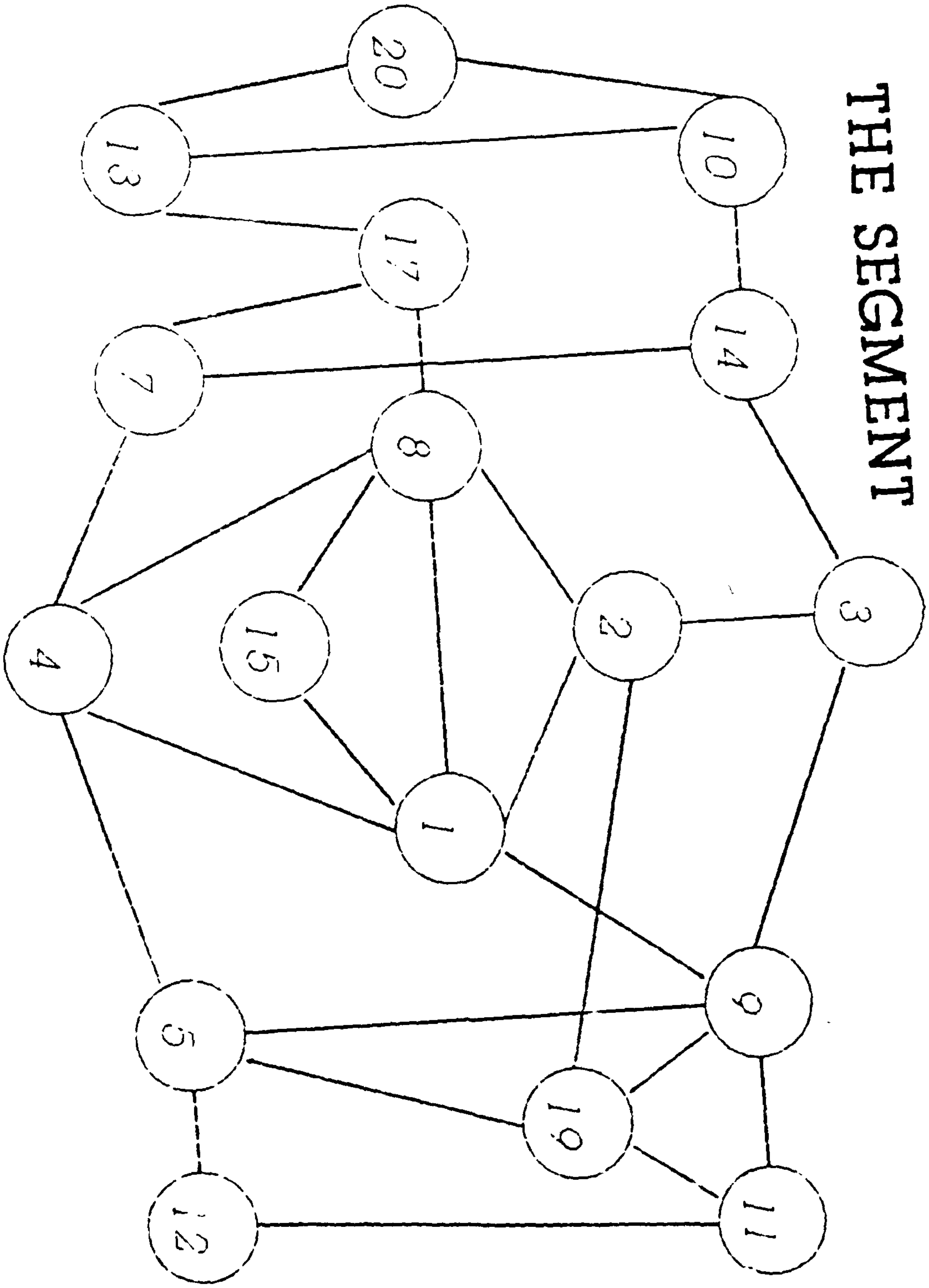
Network 1
ACC/KDC is collocated PSN1



Network 2
ACC/KDC is collacated PSN1



Network Segment
ACC/KDC is collacated PSN1



Appendix B. SLAM II Code

```
GEN,SWOPE,THESIS1,10/01/88,1,N,N,,,,72;
LIM,20,9,10000;
TIMST,XX(1),PSN1 MSG IN;
TIMST,XX(2),PSN2 MSG IN;
TIMST,XX(3),PSN3 MSG IN;
TIMST,XX(4),PSN4 MSG IN;
TIMST,XX(5),PSN5 MSG IN;
TIMST,XX(7),PSN7 MSG IN;
TIMST,XX(8),PSN8 MSG IN;
TIMST,XX(9),PSN9 MSG IN;
TIMST,XX(10),PSN10 MSG IN;
TIMST,XX(11),PSN11 MSG IN;
TIMST,XX(12),PSN12 MSG IN;
TIMST,XX(13),PSN13 MSG IN;
TIMST,XX(14),PSN14 MSG IN;
TIMST,XX(15),PSN15 MSG IN;
TIMST,XX(17),PSN17 MSG IN;
TIMST,XX(19),PSN19 MSG IN;
TIMST,XX(20),PSN20 MSG IN;
TIMST,XX(23),DUMP PACKETS;
TIMST,XX(24),LOST PACKETS;
TIMST,XX(25),SYS INPUT TOTAL;
TIMST,XX(26),DELIVERED TOTAL;
;
EQUIVALENCE/ATRIB(1),TYPE/ATRIB(2),PKTS/
            ATRIB(3),TIME/ATRIB(4),OPSN/
            ATRIB(5),DPSN/ATRIB(6),FPSN/
            ATRIB(7),CPKT/ATRIB(8),CNTR/
            ATRIB(9),HOPS;
;
PRIORITY/1,LVF(1)/6,LVF(1)/11,LVF(1)/16,LVF(1)/
         2,LVF(1)/7,LVF(1)/12,LVF(1)/17,LVF(1);
PRIORITY/3,LVF(1)/8,LVF(1)/13,LVF(1)/18,LVF(1)/
         4,LVF(1)/9,LVF(1)/14,LVF(1)/19,LVF(1)/
         5,LVF(1)/10,LVF(1)/15,LVF(1)/20,LVF(1);
;
NETWORK;
;
       CREATE,10.,,3,,1;           create routing msgs
       EVENT,5,1;
       TERM;
       CREATE,2000.,,3,,1;         create blacker rekey's
       EVENT,6,1;
       TERM;
```

```
         CREATE,EXPON(2.87,1),5.7,3,,1;  create psn1 traffic
         ASSIGN,TYPE=56.,XX(1)=XX(1)+1.,OPSN=1.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H1       EVENT,1,1;                      event at host/s
         ACT,USERF(2);                   host processing
         UNBATCH,2,1;                    unbatch host traffic
B1       EVENT,7,1;                      blacker E3
         ACT,USERF(3);                   blacker processing
PSN1     QUEUE(1),,718,BALK(LOST);       que for psn #1
         ACT,USERF(1);                   psn processing
P1       EVENT,2,1;                      event at psn #1
         ACT/1,.0000008*CPKT,FPSN.EQ.2.0,PSN2;
         ACT/2,.0000864*CPKT,FPSN.EQ.4.0,PSN4;
         ACT/3,.0000008*CPKT,FPSN.EQ.15.,PSNF;
         ACT/57,.0000008*CPKT,FPSN.EQ.8.,PSN8;
         ACT/58,.0000008*CPKT,FPSN.EQ.9.,PSN9;
         ACT,,DPSN.EQ.1.,H1;                    local host
         ACT,,,DUMP;                     dump-ack/misroutes
         CREATE,EXPON(.369,1),.37,3,,1;  create traffic psn2
         ASSIGN,TYPE=56.,XX(2)=XX(2)+1.,OPSN=2.,
              PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H2       EVENT,1,1;                      event at host/s
         ACT,USERF(2);                   host processing
         UNBATCH,2,1;                    unbatch host traffic
B2       EVENT,7,1;                      blacker E3
         ACT,USERF(3);                   blacker processing
PSN2     QUEUE(2),,718,BALK(LOST);       que for psn #2
         ACT,USERF(1);                   psn processing
P2       EVENT,2,1;                      event at psn #2
         ACT/4,.0000008*CPKT,FPSN.EQ.1.0,PSN1;
         ACT/5,.0000024*CPKT,FPSN.EQ.3.0,PSN3;
         ACT/6,.0000008*CPKT,FPSN.EQ.8.0,PSN8;
         ACT/7,.0002568*CPKT,FPSN.EQ.19.,PSNJ;
         ACT,,DPSN.EQ.2.,H2;             local host
         ACT,,,DUMP;                     dump-ack/misroutes
         CREATE,EXPON(.725,1),1.4,3,,1;  create traffic psn3
         ASSIGN,TYPE=56.,XX(3)=XX(3)+1.,OPSN=3.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H3       EVENT,1,1;                      event at host/s
         ACT,USERF(2);                   host processing
         UNBATCH,2,1;                    unbatch host traffic
B3       EVENT,7,1;                      blacker E3
         ACT,USERF(3);                   blacker processing
PSN3     QUEUE(3),,718,BALK(LOST);       que for psn #3
         ACT,USERF(1);                   psn processing
P3       EVENT,2,1;                      event at psn #3
         ACT/8,.0000024*CPKT,FPSN.EQ.2.0,PSN2;
         ACT/9,.0000008*CPKT,FPSN.EQ.9.0,PSN9;
         ACT/10,.0000096*CPKT,FPSN.EQ.14.,PSNE;
         ACT,,DPSN.EQ.3.,H3;             local host
         ACT,,,DUMP;                     dump-ack/misroutes
```

```
        CREATE,EXPON(.266,1),.5,3,,1;     create traffic psn4
        ASSIGN,TYPE=56.,XX(4)=XX(4)+1.,OPSN=4.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H4      EVENT,1,1;                        event at host/s
        ACT,USERF(2);                     host processing
        UNBATCH,2,1;                      unbatch host traffic
B4      EVENT,7,1;                        blacker E3
        ACT,USERF(3);                     blacker processing
PSN4    QUEUE(4),,718,BALK(LOST);         que for psn #4
        ACT,USERF(1);                     psn processing
        EVENT,2,1;                        event at psn #4
        ACT/11,.0000864*CPKT,FPSN.EQ.1.,PSN1;
        ACT/12,.0000064*CPKT,FPSN.EQ.5.,PSN5;
        ACT/13,.0000120*CPKT,FPSN.EQ.7.,PSN7;
        ACT/14,.0000856*CPKT,FPSN.EQ.8.,PSN8;
        ACT,,DPSN.EQ.4.,H4;               local host
        ACT,,,DUMP;                       dump-ack/misroutes
;
        CREATE,EXPON(52.6,1),5.,3,,1;     create traffic psn5
        ASSIGN,TYPE=56.,XX(5)=XX(5)+1.,OPSN=5.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H5      EVENT,1,1;                        event at host/s
        ACT,USERF(2);                     host processing
        UNBATCH,2,1;                      unbatch host traffic
B5      EVENT,7,1;                        blacker E3
        ACT,USERF(3);                     blacker processing
PSN5    QUEUE(5),,718,BALK(LOST);         que for psn #5
        ACT,USERF(1);                     psn processing
        EVENT,2,1;                        event at psn #5
        ACT/15,.0000064*CPKT,FPSN.EQ.4.0,PSN4;
        ACT/16,.0000064*CPKT,FPSN.EQ.9.0,PSN9;
        ACT/17,.0002568*CPKT,FPSN.EQ.12.,PSNC;
        ACT/59,.0002568*CPKT,FPSN.EQ.19.,PSNJ;
        ACT,,DPSN.EQ.5.,H5;               local host
        ACT,,,DUMP;                       dump-ack/misroutes
;
H6      EVENT,1,1;                        blacker ACC/KDC
        ACT,USERF(2);                     ACC/KDC processing
PSN6    QUEUE(6),,718,BALK(LOST);         que for KDC
        ACT,USERF(1);                     KDC processing
        EVENT,2,1;                        event at kdc
        ACT/51,.0000180*CPKT,FPSN.EQ.7.,PSN7;
        ACT/52,.0000012*CPKT,FPSN.EQ.8.,PSN8;
        ACT/53,.0000012*CPKT,FPSN.EQ.9.,PSN9;
        ACT,,DPSN.EQ.6.,H6;               kdc/acc/local traffic
        ACT,,,DUMP;                       dump/ack misroutes
```

```
        CREATE,EXPON(.39,1),.8,3,,1;     create traffic psn7
        ASSIGN,TYPE=56.,XX(7)=XX(7)+1.,OPSN=7.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H7      EVENT,1,1;                       event at host/s
        ACT,USERF(2);                    host processing
        UNBATCH,2,1;                     unbatch host traffic
B7      EVENT,7,1;                       blacker E3
        ACT,USERF(3);                    blacker processing
PSN7    QUEUE(7),,718,BALK(LOST);        que for psn #7
        ACT,USERF(1);                    psn processing
        EVENT,2,1;                       event at psn #7
        ACT/18,.0000120*CPKT,FPSN.EQ.4.0,PSN4;
        ACT.19,.0000120*CPKT,FPSN.EQ.14.,PSNE;
        ACT/20,.0000048*CPKT,FPSN.EQ.17.,PSNH;
        ACT/54,.0000180*CPKT,FPSN.EQ.6.0,PSN6; /acc-kdc
        ACT,,DPSN.EQ.7.,H7;              local host
        ACT,,,DUMP;                      dump-ack/misroutes
        CREATE,EXPON(.30,1),.5,3,,1;     create traffic psn8
        ASSIGN,TYPE=56.,XX(8)=XX(8)+1.,OPSN=8.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H8      EVENT,1,1;                       event at host/s
        ACT,USERF(2);                    host processing
        UNBATCH,2,1;                     unbatch host traffic
B8      EVENT,7,1;                       blacker E3
        ACT,USERF(3);                    blacker processing
PSN8    QUEUE(8),,712,BALK(LOST);        que for psn #8
        ACT,USERF(1);                    psn processing
P8      EVENT,2,1;                       event at psn #8
        ACT/21,.0000008*CPKT,FPSN.EQ.2.,PSN2;
        ACT/22,.0000856*CPKT,FPSN.EQ.4.,PSN4;
        ACT/23,.0000008*CPKT,FPSN.EQ.15.,PSNF;
        ACT/24,.0000120*CPKT,FPSN.EQ.17.,PSNH;
        ACT/55,.0000012*CPKT,FPSN.EQ.6.0,PSN6;
        ACT/60,.0000008*CPKT,FPSN.EQ.1.0,PSN1;
        ACT,,DPSN.EQ.8.,H8;              local host
        ACT,,,DUMP;                      dump-ack/misroutes
        CREATE,EXPON(1.48,1),2.9,3,,1;   create traffic psn9
        ASSIGN,TYPE=56.,XX(9)=XX(9)+1.,OPSN=9.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
H9      EVENT,1,1;                       event at host/s
        ACT,USERF(2);                    host processing
        UNBATCH,2,1;                     unbatch host traffic
B9      EVENT,7,1;                       blacker E3
        ACT,USERF(3);                    blacker processing
PSN9    QUEUE(9),,712,BALK(LOST);        que for psn #9
        ACT,USERF(1);                    psn processing
P9      EVENT,2,1;                       event at psn #9
        ACT/25,.0000008*CPKT,FPSN.EQ.3.0,PSN3;
        ACT/26,.0000064*CPKT,FPSN.EQ.5.0,PSN5;
        ACT/27,.0002568*CPKT,FPSN.EQ.11.,PSNB;
```

```
        ACT/56,.0000012*CPKT,FPSN.EQ.6.0,PSN6;
        ACT/61,.0000008*CPKT,FPSN.EQ.1.0,PSN1;
        ACT/62,.0002568*CPKT,FPSN.EQ.19.,PSNJ;
        ACT,,DPSN.EQ.9.,H9;                   local host
        ACT,,,DUMP;                           dump-ack/misroutes
        CREATE,EXPON(.52,1),.5,3,,1;          create taffic psn10
        ASSIGN,TYPE=56.,XX(10)=XX(10)+1.,OPSN=10.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HA      EVENT,1,1;                            event at host/s
        ACT,USERF(2);                         host processing
        UNBATCH,2,1;                          unbatch host traffic
BA      EVENT,7,1;                            blacker E3
        ACT,USERF(3);                         blacker processing
PSNA    QUEUE(10),,712,BALK(LOST);            que for psn #10
        ACT,USERF(1);                         psn processing
PA      EVENT,2,1;                            event at psn #10
        ACT/28,.0000008*CPKT,FPSN.EQ.13.,PSND;
        ACT/29,.0002568*CPKT,FPSN.EQ.14.,PSNE;
        ACT/30,.0002568*CPKT,FPSN.EQ.20.,PSNK;
        ACT,,DPSN.EQ.10.,HA;                  local host
        ACT,,,DUMP;                           dump-ack/misroutes
        CREATE,EXPON(1.99,1),1.9,3,,1;        create traffic psn11
        ASSIGN,TYPE=56.,XX(11)=XX(11)+1.,OPSN=11.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HB      EVENT,1,1;                            event at host/s
        ACT,USERF(2);                         host processing
        UNBATCH,2,1;                          unbatch host traffic
BB      EVENT,7,1;                            blacker E3
        ACT,USERF(3);                         blacker processing
PSNB    QUEUE(11),,712,BALK(LOST);            que for psn #11
        ACT,USERF(1);                         psn processing
PB      EVENT,2,1;                            event at psn #11
        ACT/31,.0002568*CPKT,FPSN.EQ.9.0,PSN9;
        ACT/32,.0000016*CPKT,FPSN.EQ.12.,PSNC;
        ACT/33,.0002568*CPKT,FPSN.EQ.19.,PSNJ;
        ACT,,DPSN.EQ.11.,HB;                  local host
        ACT,,,DUMP;                           dump-ack/misroutes
        CREATE,EXPON(2.13,1),2.,3,,1;         create traffic psn12
        ASSIGN,TYPE=56.,XX(12)=XX(12)+1.,OPSN=12.,
               PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HC      EVENT,1,1;                            event at host/s
        ACT,USERF(2);                         host processing
        UNBATCH,2,1;                          unbatch host traffic
BC      EVENT,7,1;                            blacker E3
        ACT,USERF(3);                         blacker processing
PSNC    QUEUE(12),,712,BALK(LOST);            que for psn #12
        ACT,USERF(1);                         psn processing
PC      EVENT,2,1;                            event at psn #12
        ACT/34,.0002568*CPKT,FPSN.EQ.5.0,PSN5;
        ACT/35,.0000016*CPKT,FPSN.EQ.11.,PSNB;
        ACT,,DPSN.EQ.12.,HC;                  local host
        ACT,,,DUMP;                           dump-ack/misroutes
```

```
          CREATE,EXPON(.298,1),.3,3,,1;    create traffic psn13
          ASSIGN,TYPE=56.,XX(13)=XX(13)+1.,OPSN=13.,
                 PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HD        EVENT,1,1;                       event at host/s
          ACT,USERF(2);                    host processing
          UNBATCH,2,1;                     unbatch host traffic
BD        EVENT,7,1;                       blacker E3
          ACT,USERF(3);                    blacker processing
PSND      QUEUE(13),,712,BALK(LOST);       que for psn #13
          ACT,USERF(1);                    psn processing
PD        EVENT,2,1;                       event at psn #13
          ACT/36,.0000008*CPKT,FPSN.EQ.10.,PSNA;
          ACT/37,.0002568*CPKT,FPSN.EQ.17.,PSNH;
          ACT/38,.0002568*CPKT,FPSN.EQ.20.,PSNK;
          ACT,,DPSN.EQ.13.,HD;             local host
          ACT,,,DUMP;                      dump-ack/misroutes
;
          CREATE,EXPON(.325,1),.6,3,,1;
          ASSIGN,TYPE=56.,XX(14)=XX(14)+1.,OPSN=14.,
                 PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HE        EVENT,1,1;                       event at host/s
          ACT,USERF(2);                    host processing
          UNBATCH,2,1;                     unbatch host traffic
BE        EVENT,7,1;                       blacker E3
          ACT,USERF(3);                    blacker processing
PSNE      QUEUE(14),,712,BALK(LOST);       que for psn #14
          ACT,USERF(1);                    psn processing
PE        EVENT,2,1;                       event at psn #14
          ACT/39,.0000096*CPKT,FPSN.EQ.3.0,PSN3;
          ACT/40,.0000120*CPKT,FPSN.EQ.7.0,PSN7;
          ACT/41,.0002568*CPKT,FPSN.EQ.10.,PSNA;
          ACT,,DPSN.EQ.14.,HE;             local host
          ACT,,,DUMP;                      dump-ack/misroutes
;
          CREATE,EXPON(.37,1),.3,3,,1;     create traffic psn15
          ASSIGN,TYPE=56.,XX(15)=XX(15)+1.,OPSN=15.,
                 PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HF        EVENT,1,1;                       event at host/s
          ACT,USERF(2);                    host processing
          UNBATCH,2,1;                     unbatch host traffic
BF        EVENT,7,1;                       blacker E3
          ACT,USERF(3);                    blacker processing
PSNF      QUEUE(15),,712,BALK(LOST);       que for psn #15
          ACT,USERF(1);                    psn processing
          EVENT,2,1;                       event at psn #15
          ACT/42,.0000008*CPKT,FPSN.EQ.1.,PSN1;
          ACT/43,.0000008*CPKT,FPSN.EQ.8.,PSN8;
          ACT,,DPSN.EQ.15.,HF;             local host
          ACT,,,DUMP;                      dump-ack/misroutes
```

```
         CREATE,EXPON(.461,1),.9,3,,1;    create traffic psn17
         ASSIGN,TYPE=56.,XX(17)=XX(17)+1.,OPSN=17.,
                PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HH       EVENT,1,1;                       event at host/s
         ACT,USERF(2);                    host processing
         UNBATCH,2,1;                     unbatch host traffic
BH       EVENT,7,1;                       blacker E3
         ACT,USERF(3);                    blacker processing
PSNH     QUEUE(17),,712,BALK(LOST);       que for psn #17
         ACT,USERF(1);                    psn processing
PH       EVENT,2,1;                       event at psn #17
         ACT/44,.0000048*CPKT,FPSN.EQ.7.0,PSN7;
         ACT/45,.0000120*CPKT,FPSN.EQ.8.0,PSN8;
         ACT/46,.0002568*CPKT,FPSN.EQ.13.,PSND;
         ACT,,DPSN.EQ.17.,HH;             local host
         ACT,,,DUMP;                      dump-ack/misroutes
;
         CREATE,EXPON(2.07,1),4.1,3,,1;   create traffic psn19
         ASSIGN,TYPE=56.,XX(19)=XX(19)+1.,OPSN=19.,
                PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HJ       EVENT,1,1;                       event at host/s
         ACT,USERF(2);                    host processing
         UNBATCH,2,1;                     unbatch host traffic
BJ       EVENT,7,1;                       blacker E3
         ACT,USERF(3);                    blacker processing
PSNJ     QUEUE(19),,712,BALK(LOST);       que for psn #19
         ACT,USERF(1);                    psn processing
PJ       EVENT,2,1;                       event at psn #19
         ACT/47,.0002568*CPKT,FPSN.EQ.2.0,PSN2;
         ACT/48,.0002568*CPKT,FPSN.EQ.11.,PSNB;
         ACT/63,.0002568*CPKT,FPSN.EQ.9.0,PSN9;
         ACT/64,.0002568*CPKT,FPSN.EQ.5.0,PSN5;
         ACT,,DPSN.EQ.19.,HJ;             local host
         ACT,,,DUMP;                      dump-ack/misroutes
;
         CREATE,EXPON(4.0,1),4.,3,,1;     create traffic psn20
         ASSIGN,TYPE=56.,XX(20)=XX(20)+1.,OPSN=20.,
                PKTS=1.,XX(25)=XX(25)+1.,CNTR=XX(25);
HK       EVENT,1,1;                       event at host/s
         ACT,USERF(2);                    host processing
         UNBATCH,2,1;                     unbatch host traffic
BK       EVENT,7,1;                       blacker E3
         ACT,USERF(3);                    blacker processing
PSNK     QUEUE(20),,712,BALK(LOST);       que for psn #20
         ACT,USERF(1);                    psn processing
PK       EVENT,2,1;                       event at psn #20
         ACT/49,.0002568*CPKT,FPSN.EQ.10.,PSNA;
         ACT/50,.0002568*CPKT,FPSN.EQ.13.,PSND;
         ACT,,DPSN.EQ.20.,HK;             local host
         ACT,,,DUMP;                      dump-ack/misroutes
;
```

```
DUMP    GOON                             acknowledgement received
        ACT,,TYPE.LE.40.,ACK;                 ist ack's
        ACT,,TYPE.EQ.50.,ACK;                 psn to host ack
        ACT,,TYPE.EQ.60.,ACK;                 host to psn ack
        ACT,,TYPE.EQ.45.,LOST;                misrouted ist packet
        ACT,,TYPE.EQ.55.,DTPK;                count data packet
        ACT,,,BAD;                            if any other ack
ACK     TERM;                                 dump it - final ack
LOST    EVENT,4,1;
BAD     ASSIGN,XX(24)=XX(24)+1.;              count misplaced pkts
NOPE    TERM;
DTPK    COLCL,INT(3),PACKET TIME,,1;
        BATCH,999/8,ATRIB(7),,LAST;           batch data packet
        EVENT,3,1;                       into msg format for host
        ASSIGN,XX(26)=XX(26)+1.;              msg thru system
        COLCT,INT(3),MESSAGE TIME,,1;
        TERM;
        END;
INIT,0.,3000.;
;MONTR,TRACE,0,10,1,2,3,4,5,6,7,8,9;
FIN;
;
;
;
;      *****************************************************
```

Appendix C. Fortran Source Code

```
      PROGRAM MAIN

      DIMENSION NSET(150000)
      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)
      COMMON QSET(150000)

      REAL A,B,RAND
      INTEGER NRN
      EQUIVALENCE(NSET(1),QSET(1))

      OPEN(UNIT=16,FILE='OUT',STATUS='NEW')
      NNSET=150000
      NCRDR=5
      NPRNT=6
      NTAPE=7
      NPLOT=2
      CALL SLAM
      STOP
      END
C
C     ****************************************************
```

```
      SUBROUTINE EVENT(I)

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1        MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2        SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)
      INTEGER I

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)
C
      GO TO (1,2,3,4,5,6,7),I
 1    CALL HOST                    ! MSG OR PACKET ARRIVED
      RETURN                       ! AT A HOST
 2    CALL SWITCH                  ! PACKET ARRIVED AT
      RETURN                       ! A SWITCH
 3    TYPE = 57.                   ! GENERATE LLD - EOM
      DPSN = PPSN                  ! RESTORE DESTINATION
      CALL HOST                    ! USE HOST SUBROUTINE
      RETURN
 4    CALL PRINT(4)                ! VALID PACKETS ARE HITTING
      RETURN                       ! THE TRASH - PRINT 'EM OUT
 5    CALL ROUTES                  ! DETERMINE CURRENT PATH
      CALL RTERS(1)                ! GENERATE ROUTING UPDATES
      RETURN
 6    CALL RTERS(2)                ! GENERATE REKEY MESSAGES
      RETURN
 7    IF (TYPE.GT.60.) CPKT = CPKT + 64. ! ADD BLACKER E3
      RETURN
      END

C
C     ************************************************
```

```
      SUBROUTINE HOST

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)
      INTEGER II

C     IF A MESSAGE IS RECEIVED FROM A TERMINAL OFF THE
C     HOST, IT WILL HAVE ATRIB(1) SET TO 56; IF IT IS
C     RECEIVED FROM THE PSN IN PACKET FORM, ATRIB(1)
C     WILL BE SET EQUAL TO A 55.

      IF(TYPE.EQ.50.) THEN   ! ACK RECEIVED FROM PSN
          TYPE = 60.         ! SET FOR DUMP/ACK
          PKTS = 1.          ! SET NBR PACKETS TO 1
          CPKT = 10.         ! SET SIZE OF PACKET
          DPSN = 0.          ! DUMP AT PSN
          RETURN
      ENDIF

      IF(TYPE.EQ.56.) THEN   ! MESSAGE IN FROM USER OFF HOST
        CALL DTDIST(1)       ! DETERMINE FINAL DESTINATION
C                GENERATE LOGICAL LINK REQUEST
          PKTS = 1.          ! SET PKTS/MSG TO 1 (MSG SIZE)
          CPKT = 52.         ! SET CHARACTERS PER PACKET
          TYPE = 61.         ! SET FOR PSN PROCESSING
C          CALL REPORT(1)    ! MESSSAGE IN FROM TRIB
          RETURN
      ENDIF

C     IF NOT LOCAL TRAFFIC - MUST COME FROM PARENT PSN
      CALL ACK(1)            ! GENERATE ACK
C                LLR IS LOGICAL LINK REQUEST

      IF(TYPE.EQ.51.) THEN   ! DATA RECEIVED FROM IST
          CALL SWAP          ! SWAP OPSN AND SPSN
          TYPE = 62.         ! SET FOR LLR ACK TO RETURN
      ENDIF
```

```
C           ACKNOWLEDGEMENT FOR LOGICAL LINK REQUEST
      IF(TYPE.EQ.52.) THEN    ! LLR ACK IN FROM FPSN
          CALL SWAP           ! SWAP OPSN AND DPSN
          TYPE = 65.          ! SET FOR DATA ON TRUNK
          CALL DTDIST(2)      ! DET NBR PACKETS IN MESSAGE
      ENDIF
C           REQUEST FOR DISCONNECT TO LOGICAL LINK
      IF(TYPE.EQ.53.) THEN    ! REQUEST FOR LLR DISCONNECT
          CALL SWAP           ! SWAP OPSN AND DPSN
          TYPE = 64.          ! SET FOR LLR DIS ACK
          PKTS = 1.           ! SET PACKETS IN MESSAGE TO 1
          CPKT = 52.          ! SET CHARS IN PACKET TO 16
      ENDIF

C              ACKNOWLEDGEMENT FOR LLD
      IF(TYPE.EQ.54.) THEN    ! ACK FOR DISCONNECT TO LLR
          PPSN = TYPE         ! SAVE PACKET TYPE
          TYPE = 60.          ! IF DISCONNECT ACK - SEND ACK
          OPSN = DPSN         ! SAVE DESP TO INPUT
          DPSN = 0.           ! ACK LLD TO PARENT PSN
          CPKT = 10.          ! ACK ONLY
      ENDIF

      IF(TYPE.EQ.55.)THEN     ! DATA PACKET
          CPKT = PKTS         ! SAVE NBR OF PACKETS
          PKTS = 1.           ! SET FOR DATA PACKET ACK
          PPSN = DPSN         ! SAVE DSN PSN
          DPSN = 0.           ! DELETE DESTINATION
      ENDIF

      IF(TYPE.EQ.57.)THEN     ! DATA OUTPUT
          CALL SWAP           ! SWAP OSRI AND DEST
          TYPE = 63.          ! GENERATE LLD
          PKTS = 1.           ! LOAD NUMBER OF PACKETS
          CPKT = 52.          ! LOAD PACKET SIZE
          II   = OPSN         ! LOAD IN INTEGER FORM
          CALL FILEM(II,ATRIB) ! STORE IN QUE FOR PROCESSING
      ENDIF
C     CALL REPORT(2)          ! RECORD HOST STATISTICS
      RETURN
      END

C
C     ****************************************************
```

```
      SUBROUTINE SWITCH

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)

      IF((DPSN.EQ.0.).AND.(TYPE.NE.47.))THEN ! INPUT ACK
         FPSN = 0.              ! GIVE NO FORWARD ADDRESS
         RETURN
      ENDIF
      IF(TYPE.GT.60.)THEN       ! INPUT DATA TRAFFIC FROM HOST
         CALL ACK(2)            ! GENERATE ACK TO HOST
        PPSN = OPSN             ! SAVE OPSN TO PREVIOUS PSN
        IF(TYPE.EQ.65.) CALL DTDIST(3) ! DET CHARACTER SIZE
        IF(DPSN.EQ.OPSN)THEN    ! IS TRAFFIC FOR THIS PSN
          TYPE = TYPE - 10.     ! DSN HERE
          FPSN = OPSN           ! STORE FOLLOW-ON PSN
          CPKT = CPKT - 64.     ! REMOVE BLACKER E3
        ELSE                    ! MUST BE TRUNK BOUND
          TYPE = TYPE - 20.     ! PREPARE FOR IST
          CALL DTFPSN           ! DETERMINE NEXT PSN
        ENDIF
        RETURN
      ENDIF

      IF((TYPE.EQ.60.).OR.(TYPE.EQ.40.).OR.
     +   (TYPE.EQ.30.).OR.(TYPE.EQ.20.))THEN    ! ACK TO PSN
C        TRUNK ACK'S GERNERATE WITH DPSN=FPSN, SO THEY
C        CLEAR THE OPSN ON OUTPUT - DUMPED AT DESTINATION
         IF(DPSN.EQ.0.)FPSN = 0. ! SET FPSN TO ZERO
          DPSN = 0.            ! SET DESTINATION TO DUMP/ACK
         RETURN
      ENDIF

     IF(TYPE.EQ.46.)THEN       ! GENERATED ROUTING UPDATE
        TYPE = 47.              ! SET FOR TRUNK TRAVEL
        FPSN = DPSN             ! LOAD NEXT PSN
        DPSN = 0.               ! ONLY TO NEXT PSN
        PPSN = OPSN             ! LOAD ORIGINATOR TO PREVIOUS
        RETURN
      ENDIF
```

```
      IF(TYPE.GT.40.)THEN      ! DATA TRAFFIC ON IST
        CALL ACK(3)            ! ACK FOR DATA PACKET
        PPSN = FPSN            ! SAVE THIS PSN TO PREVIOUS
        IF (FPSN.EQ.DPSN) THEN   ! PACKET AT DSN
           TYPE = TYPE + 10. ! PREPARE FOR DSN DELIVERY
           CPKT = CPKT - 64. ! REMOVE BLACKER E3
           RETURN              ! DELIVER TO HOST
        ELSEIF(DPSN.EQ.0.)THEN   ! ROUTING UPDATE MESSAGE
           IF(TYPE.EQ.47.)THEN   ! ONLY UPDATE ON RTG MSG
             TVL = (TNOW - ATRIB(3))*1000 ! CAL TIME
             IF (TVL.NE.0.) F(OPSN,FPSN) = TVL ! SAVE TIME
C             CALL REPORT(6) ! STATS FOR ROUTING MESSAGES
             FPSN = 0.         ! SET FOR DUMP AND ACK
             DPSN = 0.         ! NO TRUNK DESTINATION
             TYPE = 40.        ! TREAT AS TRUNK ACK
             RETURN
           ENDIF
        ELSE                   ! CONTINUE ON IST
          CALL DTFPSN          ! DETERMINE NEXT LEG
          RETURN
        ENDIF
C        CALL REPORT(5)        ! PSN TO PSN TRAFFIC
      ENDIF

      IF(TYPE.EQ.36.)THEN      ! BKDC GENERATED
        TYPE = 35.             ! PREPARE FOR TRUNK TRAVEL
        PPSN = OPSN            ! SAVE PREVIOUS PSN
        CALL DTFPSN            ! DETERMINE WHERE TO SEND IT
        RETURN                 ! LOAD IN OUTPUT QUE
      ENDIF

      IF(TYPE.EQ.35.)THEN      ! KDC REKEY MESSAGES
         CALL ACK(3)           ! SEND ACK TO PREVIOUS PSN
           IF(FPSN.EQ.DPSN)THEN  ! PACKET AT DESTINATION
                TYPE = 40.     ! DUMP AND ACK
                FPSN = 0.      ! NO FORWARDING ADDRESS
                DPSN = 0.      ! DUMP AT PARENT PSN
                RETURN
         ENDIF
         PPSN = FPSN           ! SAVE THIS PSN
         CALL DTFPSN           ! DETERMINE WHICH PSN IS NEXT
         RETURN
      ENDIF
      END

C
C     ***************************************************
```

```
      SUBROUTINE DTFPSN

       COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1        MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2        SS(100),SSL(100),TNEXT,TNOW,XX(100)
       COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
       COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
       COMMON/CONST/NRN(3),RAND(3),AC(12)

       EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +             (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +             (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +             (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +             (ATRIB(9),PPSN)

C
        FPSN = H(DPSN,PPSN)    ! LOAD FROM FORWARDING TABLE
        IF(FPSN.EQ.PPSN)FPSN = DPSN ! LAST HOP COMING UP
        RETURN
        END

C
C      ****************************************************

       SUBROUTINE REPORT(X)

       COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1        MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2        SS(100),SSL(100),TNEXT,TNOW,XX(100)
       COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
       COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
       COMMON/CONST/NRN(3),RAND(3),AC(12)

        GO TO (1,2,3,4,5,6,7)X
1       CONTINUE
2       CONTINUE                   ! OPTIONS WERE USED DURING THE
3       CONTINUE                   ! SLAM II AND FORTRAN
4       CONTINUE                   ! SUBROUTINE CODE DEVELOPMENT
5       CONTINUE                   ! TO VERTIFY AND VALIDATE
6       CONTINUE                   ! THAT MODEL WAS PERFORMING
7       CONTINUE                   ! AS INTENDED
C        WRITE(16,14)(ATRIB(I),I=1,9)  ! SNAP OUT ANY ENTRY
C 14      FORMAT(9(F8.3))              ! WITH BAD FPSN
        RETURN
        END

C
C      ****************************************************
```

```
      SUBROUTINE ACK(I)

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)
            INTEGER I,II

      GOTO (1,2,3)I
C           1  PREPARE HOST ACK TO PSN
C           2  PREPARE PSN  ACK TO HOST
C           3  PREPARE PSN  ACK TO ANOTHER PSN

1     IF(TYPE.GE.54.) RETURN ! ACK IN HOST SUBROUTINE
      AC(1) = 60.            ! GENERATE HOST ACK TO PSN
      AC(4) = ATRIB(5)       ! SENT TO YOUR PARENT PSN
      AC(5) = 0.             ! DESTINATION PSN IS IPSN
      AC(6) = 0.             ! NO NEXT/FOLLOWING PSN
      GOTO 66                ! LOAD ATRIBS, STORE IN QUE
2     AC(1) = 50.            ! GENERATE PSN ACK TO HOST
      AC(4) = OPSN           ! LOAD ORIGINATING PSN
      AC(5) = 0.             ! NO DEST PSN
      AC(6) = 0.             ! NO FOLLOWING PSN
      GOTO 66                ! FINISH IT OFF
3     AC(1) = 20.            ! GENERATE PSN TO PSN ACK
      IF (TYPE.GT.30.) AC(1)=AC(1)+10. ! ACK FOR BLACKER
      IF (TYPE.GT.40.) AC(1)=AC(1)+10.   ! ACK FOR DATA
      AC(4) = FPSN           ! LOAD THIS PSN
      AC(5) = PPSN           ! SEND IT BACK
      AC(6) = PPSN           ! DEST IS NEXT PSN
66    AC(2) = 1.             ! SET NBR OF PACKETS TO 1
      AC(3) = TNOW           ! LOAD CURRENT TIME
      AC(7) = 10.            ! SET PACKET CHARACTER SIZE
      AC(8) = ATRIB(8)       ! KEEP GLOBAL MESSAGE COUNT
      AC(9) = TYPE           ! SAVE PKT TYPE FOR SNAPS (P/O)
      II    = AC(4)          ! LOAD ACK'S INPUT QUE
C      CALL FILEM(II,AC)     ! PLACE ACK PACKET IN PSN'S QUE
      RETURN
      END

C
C     ****************************************************
```

```
      SUBROUTINE PRINT(IN)

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)
      INTEGER IN

       IF(IN.EQ.1)THEN
         WRITE(16,43)((F(I,J),J=1,10),I=1,20)
         WRITE(16,44)((F(I,J),J=11,20),I=1,20)
         WRITE(16,45)((G(I,J),J=1,10),I=1,20)
         WRITE(16,44)((G(I,J),J=11,20),I=1,20)
         WRITE(16,45)((H(I,J),J=1,10),I=1,20)
         WRITE(16,44)((H(I,J),J=11,20),I=1,20)
       ENDIF
       IF(IN.EQ.2)THEN
         WRITE(16,45)((G(I,J),J=1,10),I=1,20)
         WRITE(16,44)((G(I,J),J=11,20),I=1,20)
         WRITE(16,45)((H(I,J),J=1,10),I=1,20)
         WRITE(16,44)((H(I,J),J=11,20),I=1,20)
       ENDIF
       IF(IN.EQ.3)THEN
         WRITE(16,46)((H(I,J),J=1,20),I=1,20)
       ENDIF
       IF(IN.EQ.4)THEN
         WRITE(16,47)(ATRIB(I),I=1,9)
       ENDIF
43     FORMAT('1',1X,10(1X,F10.5))
44     FORMAT(3X,10(1X,F10.5))
45     FORMAT(1X,10(1X,F10.5))
46     FORMAT(1X,20(1X,F5.1))
47     FORMAT(9(F9.3))
       RETURN
       END

C
C      *****************************************************
```

```
      SUBROUTINE ROUTES

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)
C

1     K = 0                     ! ROW (FLOYD'S) POINTER
      DO 10 I = 1,20            ! INITIALIZE ALL ROWS
         DO 15 J = 1,20         ! INITIALIZE EACH ELEMENT
            H(I,J) = I          ! LOAD INITIAL ROW VALUE
            G(I,J) = F(I,J)     ! MOVE PSN-PSN IST TIME
            IF(G(I,J).EQ.0.) G(I,J) = 7777. ! NO ARC
15       CONTINUE
10    CONTINUE
C      CALL PRINT(1)
2     K = K + 1                 ! INCREMENT COUNTER
      DO 20 I = 1,20            ! CHECK EACH ROW
         DO 25 J = 1,20         ! CHECK EACH ELEMENT
            IF((I.NE.K).AND.(G(I,K).NE.7777.).AND.
     +         (J.NE.K).AND.(G(K,J).NE.7777.))THEN
               IF(G(I,K)+G(K,J).LT.G(I,J))THEN
                  G(I,J) = G(I,K) + G(K,J)
                  H(I,J) = H(K,J) ! QUICKER ROUTE
C                WRITE(16,23)I,J,K
C 23             FORMAT('  I = ',I3,'  J = ',I3,'  K = ',I3)
C              CALL PRINT(3)
               ENDIF
            ENDIF
25       CONTINUE
20    CONTINUE
3     IF(K.LT.20) GO TO 2
C      WRITE(16,28)TNOW
C 28    FORMAT('  TIME NOW IS   ',F20.10)
C      CALL PRINT(1)
      RETURN
      END

C
C     ****************************************************
```

```
      FUNCTION USERF(I)

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1         MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2         SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)
      INTEGER I

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)

C
      GO TO (1,2,3), I
1     IF (TYPE.LT.46.)  USERF = .0057143 ! PSN-PSN TRUNK
      IF (TYPE.EQ.50.)  USERF = 0.       ! PSN TO HOST ACK
      IF (TYPE.EQ.46.)  USERF = 0.       ! NOT PROCESSED
      IF (TYPE.EQ.36.)  USERF = 0.       ! NO TIME
      IF((TYPE.GT.50.).AND.
     +   (TYPE.LT.60.)) USERF = .00625 ! HOST-PSN-HOST
      IF((TYPE.EQ.40.).AND.
     +   (DPSN.NE.0.0)) USERF = 0.     ! ALREADY PROCESSED
      IF (TYPE.GE.60.)  USERF = .01    ! HOST-PSN TRAFFIC
      RETURN
2     USERF = .001                     ! TIME TO HOST, HOST
C   PROCESSING TIME, AND TIME BACK TO PSN (ASSUMED MINIMUM)
      RETURN
3     USERF = 0.0                      ! BLACKER PROCESS TIME
      IF (TYPE.EQ.60.) USERF = .00125 ! PROCESS ACKS
      IF (TYPE.GT.60.) USERF = .0065  ! LLR/A & LLD/A
      IF (TYPE.EQ.65.) USERF = .025   ! DATA PACKET
      RETURN
      END

C
C     ****************************************************
```

```
      SUBROUTINE RTERS(IFN)

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)

      REAL RI,RJ
      INTEGER I,II,J,IFN
C
      RI = 1.0                     ! START WITH FIRST ROW
      RJ = 1.0                     ! START WITH FIRST COLUMN

      IF(IFN.EQ.1)THEN             ! GENERATE ROUTING UPDATES
      DO 10 I = 1,20               ! VIEW ALL ROWS
         RJ    = 1.                ! RESET OUPUT COUNTER
         DO 15 J = 1,20            ! VIEW EACH ELEMENT
            IF(F(I,J).NE.0.0)THEN    ! IS THIS A NEIGHBOR
               TYPE = 46.          ! GENERATE ROUTING UPDATE
               PKTS = 1.           ! SET TO ONE PACKET
               CPKT = 21.          ! LOAD PACKET SIZE
               OPSN = RI           ! LOAD INPUT PSN
               DPSN = RJ           ! LOAD DESTINATION PSN
               CNTR = 0.           ! DESIGNATE TRUNK TRAFFIC
               IF(I.NE.J)CALL FILEM(I,ATRIB) ! LOAD IN QUE
            ENDIF
         RJ    = RJ + 1.           ! REAL CONTER FOR J
15       CONTINUE
      RI    = RI + 1.              ! REAL COUNTER FOR I
10    CONTINUE
      RETURN
      ENDIF
```

```
      IF(IFN.EQ.2)THEN          ! GENERATE REKEY MESSAGES
         RJ    = 1.             ! RESET OUPUT COUNTER
         DO 25 J = 1,20         ! START WITH FIRST COLUMN
            IF((J.EQ.6).OR.(J.EQ.16).OR.(J.EQ.18)) GOTO 24
            TYPE = 36.          ! GENERATE RRKEY MESSAGE
            PKTS = 1.           ! ONE PACKET PER MESSAGE
            CPKT = 20.          ! SET PACKET SIZE
            OPSN = 1.           ! LOAD INPUT PSN
            DPSN = RJ           ! LOAD DESTINATION
            CALL FILEM(1,ATRIB) ! LOAD IN OUTPUT QUE
24          RJ = RJ + 1.        ! REAL COUNTER FOR I
25       CONTINUE
      RETURN
      ENDIF
      END

C
C     ****************************************************

C     THE FOLLOWING SUBROUTINE IS USED TO GENERATE A RANDOM
C     NUMBER WITH AN INITIAL SEED VALUE EQUAL TO 11237. THE
C     NEW SEED VALUE
C     IS SAVED IN 'NRN' AND USED TO GENERATE THE NEXT RANDOM
C     NUMBER.  THE RANDOM NUMBER GENERATED IS STORED IN
C     'RAND' AND PASSED ON FOR USE BY OTHER SUBROUTINES.

      SUBROUTINE GENRN(X)

      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)
      INTEGER X

       IB    =   5749
       IA    =   53
       IMOD  =  32768
       IZ1   =  FLOAT(IA*NRN(X)+IB)
       NRN(X)  =  MOD(IZ1,IMOD)       ! SAVE NEW SEED VALUE
       RAND(X) =  NRN(X)/FLOAT(IMOD)  ! RAND NBR FROM 0 TO 1
       RETURN
       END

C
C      ****************************************************
```

```
C       THE FOLLOWING SUBROUTINE HAS THREE USES OR OPTIONS:
C       1.)  TO GENREATE A DESIRED OUTPUT DESTINATION CHANNEL
C            OR PACKET SWITCHING NODE;
C       2.)  TO GENERATE THE NUMBER OF PACKETS IN EACH DATA
C            MESSAGE;
C       3.)  TO GENERATE THE NUMBER OF CHARACTERS IN EACH
C            PACKET.
C            THE RANDOM NUMBER GENERATED BY THE 'GENRN'
C            SUBROUTINE IS STORED IN THE APPROPRIATE CELL
C            OF TABLE RAND BASED ON THE SUBROUTINE OPTION.

        SUBROUTINE DTDIST(IY)

       COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
     1       MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
     2       SS(100),SSL(100),TNEXT,TNOW,XX(100)
       COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
       COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
       COMMON/CONST/NRN(3),RAND(3),AC(12)

       EQUIVALENCE  (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +              (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +              (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +              (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +              (ATRIB(9),PPSN)
        REAL  Y, ATTL, TOTAL
        INTEGER IY

        CALL GENRN(IY)          ! GENERATE RANDOM NUMBER
        TOTAL = 0.              ! INITIALIZE RUNNING TOTAL
        DISP  = 1.              ! LOAD INDEX FIRST ROW
        IF (IY.EQ.1)THEN        ! DETERMINE OUTPUT PSN/HOST
           DO 101 J=1,20        ! LOOP THRU TABLE ENTRIES
             TOTAL = TOTAL + A(J,ATRIB(4)) ! ADD % TO TOTAL
             IF(RAND(1) .LE. TOTAL)THEN ! IF CUM-TOTAL GR RN
                ATRIB(5) = DISP         ! THIS IS FINAL DSN
                RETURN                  ! NBR W/I INTERVAL
             ENDIF
             DISP = DISP + 1.           ! INCREMENT TO NEXT
101        CONTINUE                     ! ADD NEXT INTERVAL
        ENDIF
        IF (IY.EQ.2)THEN       ! DETERMINE NBR PACKETS IN MSG
           DO 102 J=1,8        ! LOOP THRU MSG LENGTH TABLE
             TOTAL = TOTAL + C(J,ATRIB(4))  ! ADD % DIST
             IF(RAND(2).LE.TOTAL)THEN       ! IF EXCEEDS RN
               ATRIB(2) = DISP              ! THIS IS IT
             RETURN
             ENDIF
             DISP = DISP + 1. ! INCREMENT TO NEXT INTERVAL
102        CONTINUE
        ENDIF
```

```
          IF (IY.EQ.3)THEN          ! DETERMINE NBR CHARS IN PKT
             DO 103 J=1,64          ! LOOP THRU PACKET SIZE TABLE
               TOTAL = TOTAL + D(J,ATRIB(4)) ! ADD % DIST
               IF(RAND(3).LE.TOTAL)THEN       ! IF EXCCEDS RN
                 ATRIB(7) = (DISP * 2) + 52   ! DISP COUNT
C          16 BIT WORDS = 2 CHARS PER WORD (PLUS ADD 52 FOR
C          X.25, IP, ETC)
                 RETURN
               ENDIF
               DISP = DISP + 1. ! INCREMENT NEXT WORD INTERVAL
103          CONTINUE
          ENDIF
        RETURN
        END

C
C       *****************************************************

C         THE FOLLOWING SUBROUTINE IS USED TO SWAP THE
          ORIGINATING AND DESTINATION PACKET SWITCH NODES -
C         USED TO RETURN ACKNOWLEDGEMENTS, (DATA, LOGICAL
C         LINK REQUEST AND LOGICAL LINK DISCONNECTS)

        SUBROUTINE SWAP

        COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,II,
       1        MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,
       2        SS(100),SSL(100),TNEXT,TNOW,XX(100)
        COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
        COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
        COMMON/CONST/NRN(3),RAND(3),AC(12)

        EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
       +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
       +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
       +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
       +            (ATRIB(9),PPSN)
        REAL DUMB

          DUMB = ATRIB(4)      ! SAVE ORIGINATOR (OPSN) IN DUMB
          ATRIB(4) = ATRIB(5)! SHIFT THIS (FINAL) PSN TO OPSN
          ATRIB(5) = DUMB      ! REPLACE FINAL WITH ORIGINATOR
          ATRIB(6) = 0.        ! ZERO OUT NEXT/FOLLOW ON PSN
          ATRIB(9) = 0.        ! ZERO OUT PREVIOUS PSN
          RETURN
          END

C
C       *****************************************************
```

DATABASE FOR SERIES 1 THROUGH 4

DATABASE BELOW WAS USED FOR NETWORK 1 SERIES RUNS, IT HAS AN ADDITIONAL PSN FOR THE ACC/KDC. NEW PSN WAS NODE 6. THREE ADDITIONAL ISTS WERE INTRODUCED FROM PSN6: ONE TO PSN7, ONE TO PSN8, AND ONE TO PSN9.

```
 SUBROUTINE INTLC
 COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,
+,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),,
+TNOW,XX(100),MSTOP,TNEXT,MFA,II,NCLNR
 COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
 COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
 COMMON/CONST/NRN(3),RAND(3),AC(12)

 EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
+            (ATRIB(3),TIME),(ATRIB(4),OPSN),
+            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
+            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
+            (ATRIB(9),PPSN)

C   DATA TABLE A CONTAINS A HISTOGRAM REPRESENTATION OF A
C   PSN'S INPUT TRAFFIC MATRIX (% OF TRAFFIC FROM OPSN
C   TO DESTINATION PSN).
 DATA A /.209,.000,.000,.000,.000,.000,.000,.168,.000,.000,
+        .000,.000,.000,.000,.623,.000,.000,.000,.000,.000,
2        .000,.009,.036,.043,.011,.000,.003,.285,.064,.012,
+        .004,.005,.001,.492,.022,.000,.002,.000,.003,.008,
3        .000,.098,.147,.037,.058,.000,.006,.243,.007,.191,
+        .003,.008,.002,.055,.067,.000,.012,.000,.061,.005,
4        .000,.125,.109,.193,.008,.000,.060,.089,.118,.040,
+        .039,.009,.068,.086,.022,.000,.008,.000,.021,.005,
5        .000,.228,.019,.013,.002,.000,.016,.042,.465,.097,
+        .002,.028,.001,.063,.018,.000,.002,.000,.002,.002,
+         20*.050,
7        .000,.004,.003,.067,.090,.000,.089,.101,.122,.004,
+        .002,.076,.068,.194,.158,.000,.016,.000,.001,.005,
8        .024,.253,.021,.094,.033,.000,.016,.109,.028,.046,
+        .003,.131,.113,.105,.005,.000,.011,.000,.003,.005,
9        .000,.033,.002,.008,.009,.000,.005,.909,.002,.001,
+        .011,.005,.001,.002,.008,.000,.002,.000,.001,.001,
+        .000,.031,.069,.043,.088,.000,.010,.202,.010,.024,
+        .004,.006,.230,.003,.042,.000,.022,.000,.008,.208,
1        .000,.007,.002,.020,.001,.000,.002,.026,.854,.012,
+        .002,.050,.001,.002,.015,.000,.003,.000,.002,.001,
2        .000,.025,.022,.108,.065,.000,.073,.280,.154,.013,
+        .066,.039,.005,.078,.032,.000,.027,.000,.009,.004,
3        .000,.003,.002,.065,.002,.000,.039,.051,.002,.546,
+        .001,.003,.035,.107,.012,.000,.006,.000,.001,.125,
4        .000,.018,.005,.124,.016,.000,.364,.035,.001,.001,
```

```
     +        .001,.081,.009,.166,.009,.000,.168,.000,.001,.001,
     5        .046,.082,.083,.168,.027,.000,.101,.069,.021,.141,
     +        .116,.021,.017,.045,.004,.000,.015,.000,.034,.010,
     +          20*.050,
     7        .000,.013,.041,.021,.004,.000,.158,.187,.009,.400,
     +        .007,.013,.028,.073,.028,.000,.010,.000,.004,.004,
     +          20*.050,
     9        .000,.009,.799,.019,.002,.000,.004,.073,.005,.008,
     +        .009,.007,.003,.004,.041,.000,.004,.000,.011,.002.
     +        .000,.060,.017,.037,.007,.000,.013,.191,.011,.390,
     +        .005,.010,.181,.010,.015,.000,.009,.000,.004,.040/

C    TABLE C CONTAINS HISTOGRAM PERCENTAGE FOR NUMBER OF
C    PACKETS PER MESSAGE INPUT FROM HOSTS OFF EACH PSN.
C    PACKETS 1     2     3     4     5     6     7     8
      DATA C /1.000,.000,.000,.000,.000,.000,.000,.000,
     +          .430,.077,.010,.001,.002,.002,.000,.478,
     +          .952,.008,.002,.003,.028,.000,.000,.007,
     4          .403,.233,.033,.013,.023,.028,.001,.266,
     +          .985,.003,.000,.000,.000,.001,.000,.011,
     +          .000,.000,.000,.000,.000,.000,.000,.000,
     +          .828,.010,.003,.001,.014,.000,.000,.144,
     8          .416,.026,.082,.038,.013,.013,.016,.396,
     +          .853,.003,.001,.002,.137,.001,.000,.003,
     +          .879,.045,.005,.006,.006,.009,.000,.050,
     +          .884,.023,.056,.001,.029,.000,.000,.007,
     2          .951,.021,.001,.000,.023,.000,.000,.004,
     +          .263,.018,.001,.000,.001,.000,.018,.699,
     +          .625,.105,.007,.009,.009,.008,.001,.236,
     +          .772,.037,.024,.021,.090,.011,.001,.044,
     6          .000,.000,.000,.000,.000,.000,.000,.000,
     +          .866,.011,.039,.000,.007,.001,.001,.075,
     +          .000,.000,.000,.000,.000,.000,.000,.000,
     +          .983,.000,.000,.000,.014,.000,.000,.003,
     +          .938,.047,.001,.000,.013,.000,.000,.001/

C    TABLE D IS A HISTOGRAM REPRESENTING THE SIZE OF
C    EACH PACKET;  MAXIMUM OF 64 DATA WORDS PER PACKET,
C    64 ENTIRES PER LINE - % DIST PER WORD PER PACKET)
      DATA D / 8*.000,8*.014,16*.050,24*.003,8*.002,
     +          2*0,.001,5*0,8*.003,16*.0045,25*.028,7*.029,
     +          6*.000,2*.001,8*.012,16*.029,22*.014,10*.013,
     4          2*.001,6*0,8*.013,16*.0045,22*.026,10*.025,
     +          8*.000,8*.020,16*.044,8*.005,24*.004,
     +          64*.000,
     +          3*.001,5*.000,8*.003,16*.016,13*.023,19*.022,
     8          3*.001,3*0,2*.001,8*.002,16*.005,29*.028,3*.029,
     +          8*.000,8*.012,8*.014,8*.015,32*.021,
     +          2*.001,6*.000,6*.006,2*.007,16*.029,32*.015,
     +          5*.003,3*.000,8*.006,9*.033,7*.032,32*.013,
     2          4*.003,4*0,8*.033,16*.029,4*.009,28*.008,
     +          8*.000,8*.001,16*.002,32*.030,
     +          0,3*.001,4*.0,8*.006,11*.009,5*.010,32*.025,
```

```
+          4*.0015,3*.003,9*.002,16*.013,9*.023,23*.024,
6          64*.000,
+          8*.000,8*.005,16*.025,16*.018,16*.017,
+          64*.000,
+          8*0,8*.0145,16*.046,4*.006,12*.005,16*.004,
+          2*.002,2*.005,4*.003,2*.013,6*.014,48*.018/

C   TABLE F CONTAINS THE DELAY ON THE ARC FROM I TO J.
C   IF THE ENTRY IS ZERO, THEN NO ARC EXISTS FROM I TO J.
 DATA  F /0.,.8,0.,86.4,10*0.,.8,5*0.,
+          .8,0.,2.4,4*0.,.8,10*0.,256.8,0.,
+          0.,2.4,6*0.,.8,4*0.,9.6,6*0.,
4          86.4,3*0.,6.4,0.,12.,85.6,12*0.,
+          3*0.,6.4,4*0.,6.4,2*0.,256.8,8*0.,
+          6*0.,18.0,4.2,4.2,11*0.,
+          3*0.,12.0,0.,18.0,7*0.,12.0,0.,0.,4.8,3*0.,
8          0.,.8,0.,85.6,0.,4.2,8*0.,.8,0.,12.,3*0.,
+          0.,0.,.8,0.,6.4,4.2,4*0.,256.8,9*0.,
+          12*0.,.8,256.8,5*0.,256.8,
+          8*0.,256.8,0.,0.,1.6,6*0.,256.8,0.,
2          4*0.,256.8,5*0.,1.6,9*0.,
+          9*0.,.8,6*0.,256.8,0.,0.,256.8,
+          0.,0.,9.6,3*0.,12.,0.,0.,256.8,10*0.,
+          .8,6*0.,.8,12*0.,
6          20*0.,
+          6*0.,4.8,12.,4*0.,256.8,7*0.,
+          20*0.,
+          0.,256.8,8*0.,256.8,9*0.,
+          9*0.,256.8,0.,0.,256.8,7*0./

 DATA NRN /11237,11237,11237/  ! ORIGINAL SEED
 DATA RAND /0.0,0.0,0.0/
 CALL ROUTES
 CALL PRINT(1)
 RETURN
 END

C
C      ******************************************************
```

DATABASE FOR SERIES 5 AND 6

DATABASE BELOW WAS USED FOR NETWORK 1 SERIES RUNS WHEN THE KDC/ACC WAS COLLOCATED WITH PSN1. NO ADDITION PSNS OR ISTS WERE ADDED.

```
 SUBROUTINE INTLC
 COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,
+,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),,
+TNOW,XX(100),MSTOP,TNEXT,MFA,II,NCLNR
 COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
 COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
 COMMON/CONST/NRN(3),RAND(3),AC(12)

 EQUIVALENCE  (ATRIB(1),TYPE),(ATRIB(2),PKTS),
+             (ATRIB(3),TIME),(ATRIB(4),OPSN),
+             (ATRIB(5),DPSN),(ATRIB(6),FPSN),
+             (ATRIB(7),CPKT),(ATRIB(8),CNTR),
+             (ATRIB(9),PPSN)


C   TABLE A CONTAINS A HISTOGRAM REPRESENTATION OF A PSN'S
C   INPUT TRAFFIC MATRIX (% OF TRAFFIC TO DESTINATION PSN).
 DATA A /.209,.000,.000,.000,.000,.000,.000,.168,.000,.000,
+        .000,.000,.000,.000,.623,.000,.000,.000,.000,.000,
2        .000,.009,.036,.043,.011,.000,.003,.285,.064,.012,
+        .004,.005,.001,.492,.022,.000,.002,.000,.003,.008,
3        .000,.098,.147,.037,.058,.000,.006,.243,.007,.191,
+        .003,.008,.002,.055,.067,.000,.012,.030,.061,.005,
4        .000,.125,.109,.193,.008,.000,.060,.089,.118,.040,
+        .039,.009,.068,.086,.022,.000,.008,.000,.021,.005,
5        .000,.228,.019,.013,.002,.000,.016,.042,.465,.097,
+        .002,.028,.001,.063,.018,.000,.002,.000,.002,.002,
+         20*.050,
7        .000,.004,.003,.067,.090,.000,.089,.101,.122,.004,
+        .002,.076,.068,.194,.158,.000,.016,.000,.001,.005,
8        .024,.253,.021,.094,.033,.000,.016,.109,.028,.046,
+        .003,.131,.113,.105,.005,.000,.011,.000,.003,.005,
9        .000,.033,.002,.008,.009,.000,.005,.909,.002,.001,
+        .011,.005,.001,.002,.008,.000,.002,.000,.001,.001,
+        .000,.031,.069,.043,.088,.000,.010,.202,.010,.024,
+        .004,.006,.230,.003,.042,.000,.022,.000,.008,.208,
1        .000,.007,.002,.020,.001,.000,.002,.026,.854,.012,
+        .002,.050,.001,.002,.015,.000,.003,.000,.002,.001,
2        .000,.025,.022,.108,.065,.000,.073,.280,.154,.013,
+        .066,.039,.005,.078,.032,.000,.027,.000,.009,.004,
3        .000,.003,.002,.065,.002,.000,.039,.051,.002,.546,
+        .001,.003,.035,.107,.012,.000,.006,.000,.001,.125,
4        .000,.018,.005,.124,.016,.000,.364,.035,.001,.001,
+        .001,.081,.009,.166,.009,.000,.168,.000,.001,.001,
5        .046,.082,.083,.168,.027,.000,.101,.069,.021,.141,
+        .116,.021,.017,.045,.004,.000,.015,.000,.034,.010,
```

```
+        20*.050,
7       .000,.013,.041,.021,.004,.000,.158,.187,.009,.400,
+       .007,.013,.028,.073,.028,.000,.010,.000,.004,.004,
+        20*.050,
9       .000,.009,.799,.019,.002,.000,.004,.073,.005,.008,
+       .009,.007,.003,.004,.041,.000,.004,.000,.011,.002,
+       .000,.060,.017,.037,.007,.000,.013,.191,.011,.390,
+       .005,.010,.181,.010,.015,.000,.009,.000,.004,.040/

C     TABLE C CONTAINS HISTOGRAM PERCENTAGE FOR NUMBER OF
C     PACKETS PER MESSAGE INPUT FROM HOSTS OFF EACH PSN.
C    PACKETS 1    2    3    4    5    6    7    8
 DATA C /1.000,.000,.000,.000,.000,.000,.000,.000,
+         .430,.077,.010,.001,.002,.002,.000,.478,
+         .952,.008,.002,.003,.028,.000,.000,.007,
4         .403,.233,.033,.013,.023,.028,.001,.266,
+         .985,.003,.000,.000,.000,.001,.000,.011,
+         .000,.000,.000,.000,.000,.000,.000,.000,
+         .828,.010,.003,.001,.014,.000,.000,.144,
8         .416,.026,.082,.038,.013,.013,.016,.396,
+         .853,.003,.001,.002,.137,.001,.000,.003,
+         .879,.045,.005,.006,.006,.009,.000,.050,
+         .884,.023,.056,.001,.029,.000,.000,.007,
2         .951,.021,.001,.000,.023,.000,.000,.004,
+         .263,.018,.001,.000,.001,.000,.018,.699,
+         .625,.105,.007,.009,.009,.008,.001,.236,
+         .772,.037,.024,.021,.090,.011,.001,.044,
6         .000,.000,.000,.000,.000,.000,.000,.000,
+         .866,.011,.039,.000,.007,.001,.001,.075,
+         .000,.000,.000,.000,.000,.000,.000,.000,
+         .983,.000,.000,.000,.014,.000,.000,.003,
+         .938,.047,.001,.000,.013,.000,.000,.001/

C     TABLE D IS A HISTOGRAM REPRESENTING THE SIZE OF
C     EACH PACKET; [BMAXIMUM OF 64 DATA WORDS PER PACKET,
C     64 ENTIRES PER LINE - % DIST PER WORD PER PACKET)
 DATA D / 8*.000,8*.014,16*.050,24*.003,8*.002,
+         2*0,.001,5*0,8*.003,8*.004,8*.005,25*.028,7*.029,
+         6*.000,2*.001,8*.012,16*.029,22*.014,10*.013,
4         2*.001,6*0,8*.013,8*.004,8*.005,22*.026,10*.025,
+         8*.000,8*.020,16*.044,8*.005,24*.004,
+         64*.000,
+         3*.001,5*.000,8*.003,16*.016,13*.023,19*.022,
8         3*.001,3*0,2*.001,8*.002,16*.005,29*.028,3*.029,
+         8*.000,8*.012,8*.014,8*.015,32*.021,
+         2*.001,6*.000,6*.006,2*.007,16*.029,32*.015,
+         5*.003,3*.000,8*.006,9*.033,7*.032,32*.013,
2         4*.003,4*0,8*.033,16*.029,4*.009,28*.008,
+         8*.000,8*.001,16*.002,32*.030,
+         .000,3*.001,4*.000,8*.006,11*.009,5*.010,32*.025,
+         4*.0015,3*.003,9*.002,16*.013,9*.023,23*.024,
6         64*.000,
```

```
     +          8*.000,8*.005,16*.025,16*.018,16*.017,
     +          64*.000,
     +          8*0,8*.0145,16*.046,4*.006,12*.005,16*.004,
     +          2*.002,2*.005,4*.003,2*.013,6*.014,48*.018/

C    TABLE F CONTAINS THE DELAY ON THE ARC FROM I TO J.
C    IF THE ENTRY IS ZERO, THEN NO ARC EXISTS FROM I TO J.
      DATA  F /0.,.8,0.,86.4,10*0.,.8,5*0.,
     +          .8,0.,2.4,4*0.,.8,10*0.,256.8,0.,
     +          0.,2.4,6*0.,.8,4*0.,9.6,6*0.,
     4          86.4,3*0.,6.4,0.,12.,85.6,12*0.,
     +          3*0.,6.4,4*0.,6.4,2*0.,256.8,8*0.,
     +          6*0.,0.00,0.00,0.00,11*0.,
     +          3*0.,12.0,0.,0.00,7*0.,12.0,0.,0.,4.8,3*0.,
     8          0.,.8,0.,85.6,0.,0.00,8*0.,.8,0.,12.,3*0.,
     +          0.,0.,.8,0.,6.4,0.00,4*0.,256.8,9*0.,
     +          12*0.,.8,256.8,5*0.,256.8,
     +          8*0.,256.8,0.,0.,1.6,6*0.,256.8,0.,
     2          4*0.,256.8,5*0.,1.6,9*0.,
     +          9*0.,.8,6*0.,256.8,0.,0.,256.8,
     +          0.,0.,9.6,3*0.,12.,0.,0.,256.8,10*0.,
     +          .8,6*0.,.8,12*0.,
     6          20*0.,
     +          6*0.,4.8,12.,4*0.,256.8,7*0.,
     +          20*0.,
     +          0.,256.8,8*0.,256.8,9*0.,
     +          9*0.,256.8,0.,0.,256.8,7*0./

      DATA NRN /11237,11237,11237/   ! ORIGINAL SEED
      DATA RAND /0.0,0.0,0.0/
      CALL ROUTES
C         CALL PRINT(1)
      RETURN
      END

C
C         ****************************************************
```

DATABASE FOR SERIES 7 AND 8

DATABASE BELOW WAS USED FOR NETWORK 2 SERIES RUNS. THE KDC/ACC IS COLLOCATED AT PSN1. NO ADDITIONAL PSNS OR ISTS WERE ADDED.

```
      SUBROUTINE INTLC
      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,
     +,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),,
     +TNOW,XX(100),MSTOP,TNEXT,MFA,II,NCLNR
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)

C     TABLE A CONTAINS A HISTOGRAM REPRESENTATION OF A PSN'S
C     INPUT TRAFFIC MATRIX (% OF TRAFFIC TO DESTINATION PSN).
      DATA A /.1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     +        20*.05,
     3        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     4        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     5        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     +        20*.05,
     7        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     8        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     9        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     +        20*.05,
     +        20*.05,
     +        20*.05,
     +        20*.05,
     4        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     +        20*.05,
     +        20*.05,
     7        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     +        20*.05,
     9        .1,0.,3*.1,.0,3*.1,4*0.,.1,2*0.,.1,0.,.1,0.,
     +        20*.05/
```

```
C     TABLE C CONTAINS HISTOGRAM PERCENTAGE FOR NUMBER OF
C     PACKETS PER MESSAGE INPUT FROM HOSTS OFF EACH PSN.
C   PACKETS  1    2    3    4    5    6    7    8
 DATA C /1.000,.000,.000,.000,.000,.000,.000,.000,
+          .430,.077,.010,.001,.002,.002,.000,.478,
+          .952,.008,.002,.003,.028,.000,.000,.007,
4          .403,.233,.033,.013,.023,.028,.001,.266,
+          .985,.003,.000,.000,.000,.001,.000,.011,
+          .000,.000,.000,.000,.000,.000,.000,.000,
+          .828,.010,.003,.001,.014,.000,.000,.144,
8          .416,.026,.082,.038,.013,.013,.016,.396,
+          .853,.003,.001,.002,.137,.001,.000,.003,
+          .879,.045,.005,.006,.006,.009,.000,.050,
+          .884,.023,.056,.001,.029,.000,.000,.007,
2          .951,.021,.001,.000,.023,.000,.000,.004,
+          .263,.018,.001,.000,.001,.000,.018,.699,
+          .625,.105,.007,.009,.009,.008,.001,.236,
+          .772,.037,.024,.021,.090,.011,.001,.044,
6          .000,.000,.000,.000,.000,.000,.000,.000,
+          .866,.011,.039,.000,.007,.001,.001,.075,
+          .000,.000,.000,.000,.000,.000,.000,.000,
+          .983,.000,.000,.000,.014,.000,.000,.003,
+          .938,.047,.001,.000,.013,.000,.000,.001/

C     TABLE D IS A HISTOGRAM REPRESENTING THE SIZE OF
C     EACH PACKET; [BMAXIMUM OF 64 DATA WORDS PER PACKET,
C     64 ENTIRES PER LINE - % DIST PER WORD PER PACKET)
 DATA D / 8*.000,8*.014,16*.050,24*.003,8*.002,
+         2*0,.001,5*0,16*.0035,8*.005,25*.028,7*.029,
+         6*0,2*.001,8*.012,16*.029,22*.014,10*.013,
4         2*.001,6*0,8*.013,16*.0045,22*.026,10*.025,
+         8*.000,8*.020,16*.044,8*.005,24*.004,
+         64*.000,
+         3*.001,5*.000,8*.003,16*.016,13*.023,19*.022,
8         3*.001,3*0,2*.001,8*.002,16*.005,29*.028,3*.029,
+         8*.000,8*.012,8*.014,8*.015,32*.021,
+         2*.001,6*.000,6*.006,2*.007,16*.029,32*.015,
+         5*.003,3*.000,8*.006,9*.033,7*.032,32*.013,
2         4*.003,4*0,8*.033,16*.029,4*.009,28*.008,
+         8*.000,8*.001,16*.002,32*.030,
+         .000,3*.001,4*.000,8*.006,11*.009,5*.010,32*.025,
+         4*.0015,3*.003,9*.002,16*.013,9*.023,23*.024,
6         64*.000,
+         8*.000,8*.005,16*.025,16*.018,16*.017,
+         64*.000,
+         8*0,8*.0145,16*.046,4*.006,12*.005,16*.004,
+         2*.002,2*.005,4*.003,2*.013,6*.014,48*.018/
```

```
C    TABLE F CONTAINS THE DELAY ON THE ARC FROM I TO J.
C    IF THE ENTRY IS ZERO, THEN NO ARC EXISTS FROM I TO J.
      DATA  F /3*0.,86.4,3*0.,.8,.8,11*0.,
     +         20*0.0,
     3         8*0.,2.4,4*0.,9.6,6*.0,
     4         86.4,3*0.,6.4,0.,12.,85.6,12*0.,
     +         3*0.,6.4,4*0.,6.4,9*0.,256.8,0.,
     +         20*0.,
     +         3*0.,12.0,0.,0.00,7*0.,12.0,0.,0.,4.8,3*0.,
     8         .8,.00,0.,85.6,12*0.,12.,3*0.,
     +         .8,0.,2.4,0.,6.4,13*0.,256.8,0.0,
     +         20*0.0,
     +         20*0.0,
     +         20*0.0,
     +         20*0.0,
     4         2*0.,9.6,3*0.,12.0,13*0.,
     +         20*0.0,
     6         20*0.,
     +         6*0.,4.8,12.,4*0.,0.0,7*0.,
     +         20*0.,
     +         4*0.0,256.8,3*0.0,256.8,11*0.0,
     +         20*0.0/

      DATA NRN /11237,11237,11237/   ! ORIGINAL SEED
      DATA RAND /0.0,0.0,0.0/
      CALL ROUTES
C         CALL PRINT(1)
      RETURN
      END

C
C      ******************************************************
```

DATABASE FOR SERIES 9

THE FOLLWOING DATABASE WAS USE FOR THE SEGMENT NETWORK. THE SEGMENT IS CONSTRUCTED BY COMBINING NETWORKS 1 AND 2, IT DOES NOT CONTAIN AN ADDITIONAL PSN FOR THE KDC/ACC. ANY ISTS AND PSNS THAT ARE IN BOTH NETWORKS 1 AND 2 ARE LISTED ONLY ONCE BELOW.

```
 SUBROUTINE INTLC
 COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,
+,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),,
+TNOW,XX(100),MSTOP,TNEXT,MFA,II,NCLNR
 COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
 COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
 COMMON/CONST/NRN(3),RAND(3),AC(12)

 EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
+            (ATRIB(3),TIME),(ATRIB(4),OPSN),
+            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
+            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
+            (ATRIB(9),PPSN)

C    TABLE A CONTAINS A HISTOGRAM REPRESENTATION OF A PSN'S
C    INPUT TRAFFIC MATRIX (% OF TRAFFIC TO DESTINATION PSN).
 DATA A /.155,.000,.050,.050,.050,.000,.050,.134,.050,.000,
+       .000,.000,.000,.050,.311,.000,.050,.000,.050,.000,
2       .000,.009,.036,.043,.011,.000,.003,.285,.064,.012,
+       .004,.005,.001,.492,.022,.000,.002,.000,.003,.008,
3       .050,.048,.124,.069,.079,.000,.053,.171,.054,.096,
+       .001,.004,.001,.078,.033,.000,.056,.000,.080,.003,
4       .050,.063,.104,.146,.054,.000,.080,.095,.109,.020,
+       .020,.004,.034,.093,.011,.000,.054,.000,.060,.003,
5       .050,.114,.059,.056,.051,.000,.058,.071,.283,.048,
+       .001,.014,.001,.082,.009,.000,.051,.000,.051,.001,
+        20*.050,
7       .050,.002,.051,.084,.095,.000,.095,.101,.111,.002,
+       .001,.038,.034,.147,.079,.000,.058,.000,.050,.002,
8       .063,.126,.061,.097,.067,.000,.058,.104,.064,.023,
+       .002,.065,.056,.102,.003,.000,.055,.000,.051,.003,
9       .050,.017,.051,.054,.054,.000,.052,.504,.051,.001,
+       .006,.002,.001,.051,.004,.000,.051,.000,.050,.001,
+       .000,.031,.069,.043,.088,.000,.010,.202,.010,.024,
+       .004,.006,.230,.003,.042,.000,.022,.000,.008,.208,
1       .000,.007,.002,.020,.001,.000,.002,.026,.854,.012,
+       .002,.050,.001,.002,.015,.000,.003,.000,.002,.001,
2       .000,.025,.022,.108,.065,.000,.073,.280,.154,.013,
+       .066,.039,.005,.078,.032,.000,.027,.000,.009,.004,
3       .000,.003,.002,.065,.002,.000,.039,.051,.002,.546,
+       .001,.003,.035,.107,.012,.000,.006,.000,.001,.125,
4       .051,.009,.052,.112,.058,.000,.232,.067,.050,.001,
+       .001,.040,.004,.133,.004,.000,.134,.000,.051,.001,
5       .046,.082,.083,.168,.027,.000,.101,.069,.021,.141,
```

```
+       .116,.021,.017,.045,.004,.000,.015,.000,.034,.010,
+         20*.050,
7       .050,.007,.071,.061,.052,.000,.129,.143,.054,.200,
+       .003,.007,.014,.086,.014,.000,.055,.000,.052,.002,
+         20*.050,
9       .050,.005,.449,.059,.051,.000,.052,.086,.053,.004,
+       .005,.003,.002,.052,.021,.000,.052,.000,.055,.001,
+       .000,.060,.017,.037,.007,.000,.013,.191,.011,.390,
+       .005,.010,.181,.010,.015,.000,.009,.000,.004,.040/

C   TABLE C CONTAINS HISTOGRAM PERCENTAGE FOR NUMBER OF
C   PACKETS PER MESSAGE INPUT FROM HOSTS OFF EACH PSN.
C  PACKETS  1    2    3    4    5    6    7    8
 DATA C /1.000,.000,.000,.000,.000,.000,.000,.000,
+        .430,.077,.010,.001,.002,.002,.000,.478,
+        .952,.008,.002,.003,.028,.000,.000,.007,
4        .403,.233,.033,.013,.023,.028,.001,.266,
+        .985,.003,.000,.000,.000,.001,.000,.011,
+        .000,.000,.000,.000,.000,.000,.000,.000,
+        .828,.010,.003,.001,.014,.000,.000,.144,
8        .416,.026,.082,.038,.013,.013,.016,.396,
+        .853,.003,.001,.002,.137,.001,.000,.003,
+        .879,.045,.005,.006,.006,.009,.000,.050,
+        .884,.023,.056,.001,.029,.000,.000,.007,
2        .951,.021,.001,.000,.023,.000,.000,.004,
+        .263,.018,.001,.000,.001,.000,.018,.699,
+        .625,.105,.007,.009,.009,.008,.001,.236,
+        .772,.037,.024,.021,.090,.011,.001,.044,
6        .000,.000,.000,.000,.000,.000,.000,.000,
+        .866,.011,.039,.000,.007,.001,.001,.075,
+        .000,.000,.000,.000,.000,.000,.000,.000,
+        .983,.000,.000,.000,.014,.000,.000,.003,
+        .938,.047,.001,.000,.013,.000,.000,.001/


C   TABLE D IS A HISTOGRAM REPRESENTING THE SIZE OF
C   EACH PACKET; [BMAXIMUM OF 64 DATA WORDS PER PACKET,
C   64 ENTIRES PER LINE - % DIST PER WORD PER PACKET)
 DATA D / 8*.000,8*.014,16*.050,24*.003,8*.002,
+        2*0,.001,5*0,16*.0035,8*.005,25*.028,7*.029,
+        6*0,2*.001,8*.012,16*.029,22*.014,10*.013,
4        2*.001,6*0,8*.013,16*.0045,22*.026,10*.025,
+        8*.000,8*.020,16*.044,8*.005,24*.004,
+        64*.000,
+        3*.001,5*0,8*.003,16*.016,13*.023,19*.022,
8        6*.0005,2*.001,8*.002,16*.005,29*.028,3*.029,
+        8*0,8*.012,8*.014,8*.015,32*.021,
+        2*.001,6*0,6*.006,2*.007,16*.029,32*.015,
+        5*.003,3*0,8*.006,9*.033,7*.032,32*.013,
2        4*.003,4*0,8*.033,16*.029,4*.009,28*.008,
+        8*0,8*.001,16*.002,32*.030,
+        0,3*.001,4*0,8*.006,11*.009,5*.010,32*.025,
+        4*.0015,3*.003,9*.002,16*.013,9*.023,23*.024,
```

```
6          64*.000,
+          8*0,8*.005,16*.025,16*.018,16*.017,
+          64*.000,
+          8*0,8*.0145,16*.046,4*.006,12*.005,16*.004,
+          2*.002,2*.005,4*.003,2*.013,6*.014,48*.018/


C   TABLE F CONTAINS THE DELAY ON THE ARC FROM I TO J.
C   IF THE ENTRY IS ZERO, THEN NO ARC EXISTS FROM I TO J.
 DATA  F /0.,.8,0.,86.4,3*0.,.8,.8,5*0.,.8,5*0.,
+          .8,0.,2.4,4*0.,.8,10*0.,256.8,0.,
+          0.,2.4,6*0.,.8,4*0.,9.6,6*0.,
4          86.4,3*0.,6.4,0.,12.,85.6,12*0.,
+          3*0.,6.4,4*0.,6.4,2*0.,256.8,6*0.,256.8,0.,
+          6*0.,0.00,0.00,0.00,11*0.,
+          3*0.,12.0,0.,0.00,7*0.,12.0,0.,0.,4.8,3*0.,
8          .8,.8,0.,85.6,0.,0.00,8*0.,.8,0.,12.,3*0.,
+          .8,0.,.8,0.,6.4,0.00,4*0.,256.8,7*0.,256.8,0.,
+          12*0.,.8,256.8,5*0.,256.8,
+          8*0.,256.8,0.,0.,1.6,6*0.,256.8,0.,
2          4*0.,256.8,5*0.,1.6,9*0.,
+          9*0.,.8,6*0.,256.8,0.,0.,256.8,
+          0.,0.,9.6,3*0.,12.,0.,0.,256.8,10*0.,
+          .8,6*0.,.8,12*0.,
6          20*0.,
+          6*0.,4.8,12.,4*0.,256.8,7*0.,
+          20*0.,
+          0.,256.8,2*0.,256.8,3*0.,256.8,0.,256.8,9*0.,
+          9*0.,256.8,0.,0.,256.8,7*0./

 DATA NRN /11237,11237,11237/   ! ORIGINAL SEED
 DATA RAND /0.0,0.0,0.0/
 CALL ROUTES
C     CALL PRINT(1)
 RETURN
 END

C
C     *****************************************************
```

Appendix E. Routing Algorithm Example

Copy of Subroutine ROUTES used on Sample Network

```
C ***********************************************************
      SUBROUTINE ROUTES
      COMMON/SCOM1/ATRIB(100),DD(100),DDL(100),DTNOW,
     +,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),,
     +TNOW,XX(100),MSTOP,TNEXT,MFA,II,NCLNR
      COMMON/TABLES/A(20,20),B(20,20),C(8,20),D(64,20)
      COMMON/FLOYD/F(20,20),G(20,20),H(20,20)
      COMMON/CONST/NRN(3),RAND(3),AC(12)

      EQUIVALENCE (ATRIB(1),TYPE),(ATRIB(2),PKTS),
     +            (ATRIB(3),TIME),(ATRIB(4),OPSN),
     +            (ATRIB(5),DPSN),(ATRIB(6),FPSN),
     +            (ATRIB(7),CPKT),(ATRIB(8),CNTR),
     +            (ATRIB(9),PPSN)
C
1     K = 0                          ! ROW (FLOYD'S) POINTER
      DO 10 I = 1,20                 ! INITIALIZE ALL ROWS
         DO 15 J = 1,20              ! INITIALIZE EACH ELEMENT
            H(I,J) = I               ! LOAD INITIAL ROW VALUE
            G(I,J) = F(I,J)          ! MOVE PSN-PSN IST TIME
            IF(G(I,J).EQ.0.) G(I,J) = 7777. ! NO ARC-LARGE #
15       CONTINUE                    ! LOOP THRU 20 COLUMNS
10    CONTINUE                       ! CONTINUE WITH NEXT ROW
      CALL PRINT                     ! PRINT OUT TABLES
2     K = K + 1                      ! INCREMENT COUNTER
      DO 20 I = 1,20                 ! CHECK EACH ROW
         DO 25 J = 1,20              ! CHECK EACH ELEMENT
            IF((I.NE.K).AND.(G(I,K).NE.7777.))
            IF((J.NE.K).AND.(G(K,J).NE.7777.))THEN
              IF(G(I,K)+G(K,J).LT.G(I,J))THEN
                G(I,J) = G(I,K) + G(K,J)
                H(I,J) = H(K,J)        ! QUICKER ROUTE
              ENDIF
            ENDIF
25      CONTINUE                      ! MORE COLUMNS
20    CONTINUE                        ! MORE ROWS
      IF(K.LT.6)CALL PRINT            ! PRINT OUT TABLE
3     IF(K.LT.20) GO TO 2             ! NOT ALL ITERATIONS
      RETURN                          ! FINISHED - RETURN
      END
C
C ***********************************************************
```

The following changes were made to subroutine INTLC. The values of table F were modified to reflect the adjacency matrix for the sample network, Figure 2 Chapter II. Value F(i,j) contained the time delay from node i to node j. If no arc existed from node i to node j, then F(i,j) was set equal to zero.

Subroutine PRINT was modified to print out only the first five entries of the first five rows of each table. All the remaining entries of Tables F, G, and H contained zeros.

All other tables of Subroutine INTLC not listed below were not modified.

```
C ************************************************************

      SUBROUTINE INTLC

      DATA   F /0.,0.,2.,0.,4.,15*0.,
     +          1.,0.,1.,0.,2.,15*0.,
     +          4.,3.,0.,3.,0.,15*0.,
     +          0.,4.,0.,0.,1.,15*0.,
     +          0.,0.,5.,4.,0.,15*0.,
     +          300*0./

       DATA NRN /11237,11237,11237/   ! ORIGINAL SEED
                                      ! VALUE/RANDOM NUMBER

      DATA RAND /0.0,0.0,0.0/

      CALL ROUTES
      RETURN
      END
C
C     ******************************************************
```

Table G was used as a work area, and entry G(i,j) contained the current calculated cost from i to j. If an arc from i to j existed when ROUTES was initially called, then entry G(i,j) was loaded with that cost or time delay. If no arc existed from i to j, then G(i,j) was loaded with the value 7777 when ROUTES was initially called. Table H(i,j) contained the next node from i on the journey from i to j. As the iterations proceed, G(i,j) contained the calculated time delay from i to j using H(i,j) as the next hop of the journey. K below is used to indicate the i-th iteration or loop through ROUTES.

****

Iteration One K = 1

TABLE F

0.00E+00	0.10E+01	0.40E+01	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.30E+01	0.40E+01	0.00E+00
0.20E+01	0.10E+01	0.00E+00	0.00E+00	0.50E+01
0.00E+00	0.00E+00	0.30E+01	0.00E+00	0.40E+01
0.40E+01	0.20E+01	0.00E+00	0.10E+01	0.00E+00

TABLE G

0.78E+04	0.10E+01	0.40E+01	0.78E+04	0.78E+04
0.78E+04	0.78E+04	0.30E+01	0.40E+01	0.78E+04
0.20E+01	0.10E+01	0.78E+04	0.78E+04	0.50E+01
0.78E+04	0.78E+04	0.30E+01	0.78E+04	0.40E+01
0.40E+01	0.20E+01	0.78E+04	0.10E+01	0.78E+04

TABLE H

1.00	1.00	1.00	1.00	1.00
2.00	2.00	2.00	2.00	2.00
3.00	3.00	3.00	3.00	3.00
4.00	4.00	4.00	4.00	4.00
5.00	5.00	5.00	5.00	5.00

Iteration Two K = 2

TABLE F

0.00E+00	0.10E+01	0.40E+01	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.30E+01	0.40E+01	0.00E+00
0.20E+01	0.10E+01	0.00E+00	0.00E+00	0.50E+01
0.00E+00	0.00E+00	0.30E+01	0.00E+00	0.40E+01
0.40E+01	0.20E+01	0.00E+00	0.10E+01	0.00E+00

TABLE G

0.78E+04	0.10E+01	0.40E+01	0.78E+04	0.78E+04
0.78E+04	0.78E+04	0.30E+01	0.40E+01	0.78E+04
0.20E+01	0.10E+01	0.60E+01	0.78E+04	0.50E+01
0.78E+04	0.78E+04	0.30E+01	0.78E+04	0.40E+01
0.40E+01	0.20E+01	0.80E+01	0.10E+01	0.78E+04

TABLE H

1.00	1.00	1.00	1.00	1.00
2.00	2.00	2.00	2.00	2.00
3.00	3.00	1.00	3.00	3.00
4.00	4.00	4.00	4.00	4.00
5.00	5.00	1.00	5.00	5.00

Iteration Three K = 3

TABLE F

0.00E+00	0.10E+01	0.40E+01	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.30E+01	0.40E+01	0.00E+00
0.20E+01	0.10E+01	0.00E+00	0.00E+00	0.50E+01
0.00E+00	0.00E+00	0.30E+01	0.00E+00	0.40E+01
0.40E+01	0.20E+01	0.00E+00	0.10E+01	0.00E+00

TABLE G

0.78E+04	0.10E+01	0.40E+01	0.50E+01	0.78E+04
0.78E+04	0.78E+04	0.30E+01	0.40E+01	0.78E+04
0.20E+01	0.10E+01	0.40E+01	0.50E+01	0.50E+01
0.78E+04	0.78E+04	0.30E+01	0.78E+04	0.40E+01
0.40E+01	0.20E+01	0.50E+01	0.10E+01	0.78E+04

TABLE H

1.00	1.00	1.00	2.00	1.00
2.00	2.00	2.00	2.00	2.00
3.00	3.00	2.00	2.00	3.00
4.00	4.00	4.00	4.00	4.00
5.00	5.00	2.00	5.00	5.00

Iteration Four K = 4

TABLE F

0.00E+00	0.10E+01	0.40E+01	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.30E+01	0.40E+01	0.00E+00
0.20E+01	0.10E+01	0.00E+00	0.00E+00	0.50E+01
0.00E+00	0.00E+00	0.30E+01	0.00E+00	0.40E+01
0.40E+01	0.20E+01	0.00E+00	0.10E+01	0.00E+00

TABLE G

0.60E+01	0.10E+01	0.40E+01	0.50E+01	0.90E+01
0.50E+01	0.40E+01	0.30E+01	0.40E+01	0.80E+01
0.20E+01	0.10E+01	0.40E+01	0.50E+01	0.50E+01
0.50E+01	0.40E+01	0.30E+01	0.80E+01	0.40E+01
0.40E+01	0.20E+01	0.50E+01	0.10E+01	0.10E+02

TABLE H

3.00	1.00	1.00	2.00	3.00
3.00	3.00	2.00	2.00	3.00
3.00	3.00	2.00	2.00	3.00
3.00	3.00	4.00	2.00	4.00
5.00	5.00	2.00	5.00	3.00

Iteration Five K = 5

TABLE F

0.00E+00	0.10E+01	0.40E+01	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.30E+01	0.40E+01	0.00E+00
0.20E+01	0.10E+01	0.00E+00	0.00E+00	0.50E+01
0.00E+00	0.00E+00	0.30E+01	0.00E+00	0.40E+01
0.40E+01	0.20E+01	0.00E+00	0.10E+01	0.00E+00

TABLE G

0.60E+01	0.10E+01	0.40E+01	0.50E+01	0.90E+01
0.50E+01	0.40E+01	0.30E+01	0.40E+01	0.80E+01
0.20E+01	0.10E+01	0.40E+01	0.50E+01	0.50E+01
0.50E+01	0.40E+01	0.30E+01	0.80E+01	0.40E+01
0.40E+01	0.20E+01	0.40E+01	0.10E+01	0.50E+01

TABLE H

3.00	1.00	1.00	2.00	3.00
3.00	3.00	2.00	2.00	3.00
3.00	3.00	2.00	2.00	3.00
3.00	3.00	4.00	2.00	4.00
5.00	5.00	4.00	5.00	4.00

Iteration Six K = 6

TABLE F

0.00E+00	0.10E+01	0.40E+01	0.00E+00	0.00E+00
0.00E+00	0.00E+00	0.30E+01	0.40E+01	0.00E+00
0.20E+01	0.10E+01	0.00E+00	0.00E+00	0.50E+01
0.00E+00	0.00E+00	0.30E+01	0.00E+00	0.40E+01
0.40E+01	0.20E+01	0.00E+00	0.10E+01	0.00E+00

TABLE G

0.60E+01	0.10E+01	0.40E+01	0.50E+01	0.90E+01
0.50E+01	0.40E+01	0.30E+01	0.40E+01	0.80E+01
0.20E+01	0.10E+01	0.40E+01	0.50E+01	0.50E+01
0.50E+01	0.40E+01	0.30E+01	0.50E+01	0.40E+01
0.40E+01	0.20E+01	0.40E+01	0.10E+01	0.50E+01

TABLE H

3.00	1.00	1.00	2.00	3.00
3.00	3.00	2.00	2.00	3.00
3.00	3.00	2.00	2.00	3.00
3.00	3.00	4.00	5.00	4.00
5.00	5.00	4.00	5.00	4.00

****

Tables G and H can easily be validated by tracing the possible paths through the network, see Figure 2 Chapter II. Entry (i,j) where i is equal to j can be ignored, there is no cost from i to itself and there is no next hop in the journey from i to i.

For example, consider i=1 and j=4. Entry H(1,4) has a value of 2, the next hop is node 2. Entry H(2,4) has a value of 2, therefore 2 is the last hop before node 4. Entry G(1,4) has a value of 5, total delay from node 1 to node 4. That value is confirmed by adding the delay on each leg of the journey. There was a delay of 1 from node 1 to node 2, and a delay of 4 from node 2 to node 4. All other paths and distances can be confirmed in a similar manner.

Appendix F. Regression of Input Model Parameters

Matrix used in the Plackett-Burman design with N=20.

OBS	A1	A2	A3	A4	A5	A7	A8	A9	A10	A11	A12	A13	A14	A15	A17	A19	A20	C	D	Y1
1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	0.685
2	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	0.631
3	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	0.701
4	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	0.610
5	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	0.614
6	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	0.597
7	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	0.708
8	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	1	0.626
9	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	-1	0.741
10	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	1	0.654
11	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	-1	0.712
12	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	1	0.684
13	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	1	0.694
14	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	0.693
15	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	0.609
16	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	0.635
17	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	-1	0.742
18	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	1	0.697
19	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	1	0.624
20	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0.638

REGRESSION FOR FULL MODEL

DEP VARIABLE: Y1
ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	19	0.03996175	0.00210325	.	.
ERROR	0	0	.		
C TOTAL	19	0.03996175			

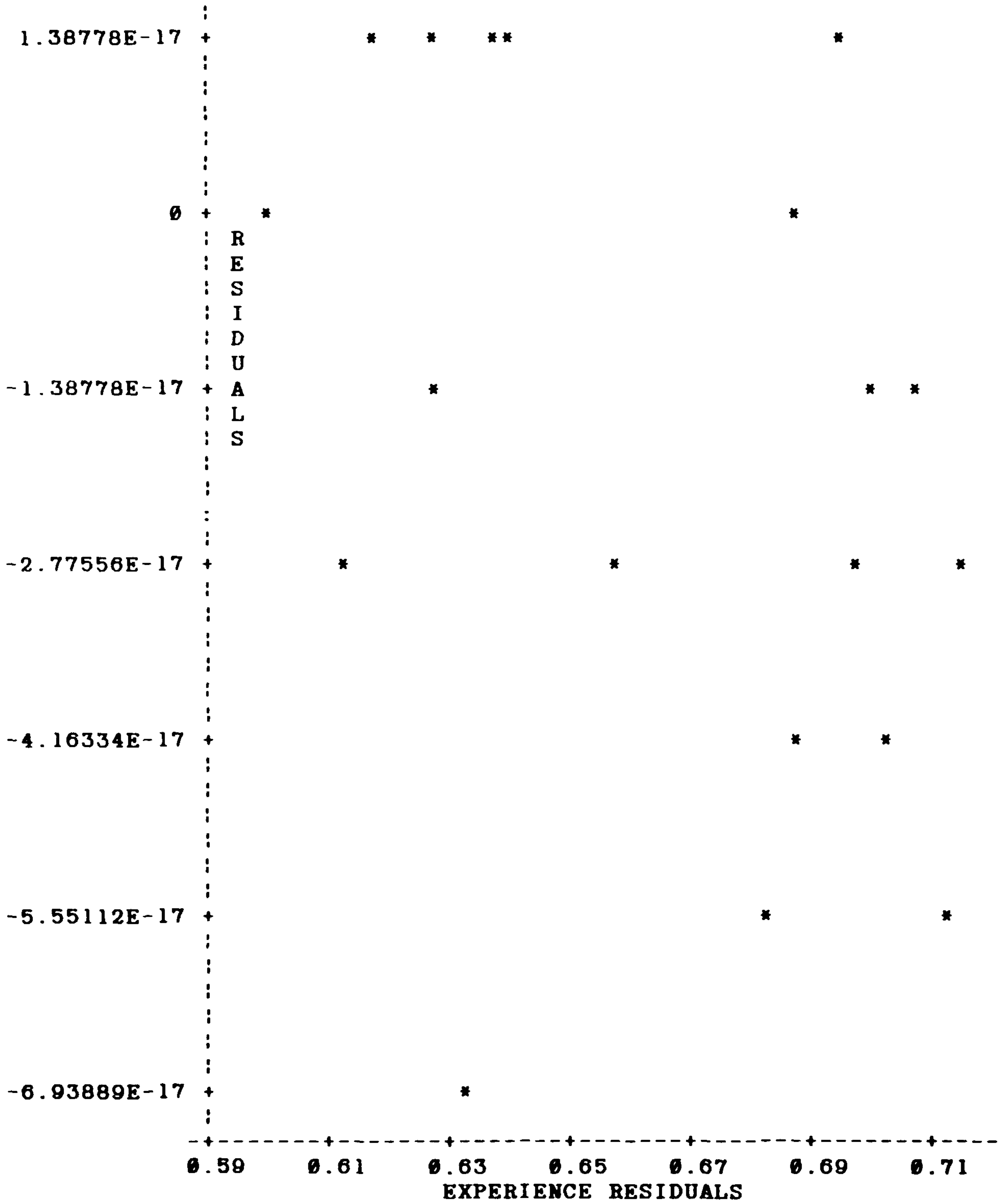
ROOT MSE	.	R-SQUARE	1.0000
DEP MEAN	0.66475	ADJ R-SQ	.
C.V.	.		

PARAMETER ESTIMATES

VARIABLE	DF	ESTIMATE
INTERCEP	1	0.66475000
A1	1	0.00565
A2	1	-0.00905
A3	1	-0.00155
A4	1	0.00435
A5	1	-0.00385
A7	1	-0.01045
A8	1	0.00445
A9	1	0.00115
A10	1	0.00375
A11	1	0.00295
A12	1	-0.01105
A13	1	-0.00185
A14	1	-0.01125
A15	1	0.01605
A17	1	0.00335
A19	1	0.00305
A20	1	0.00055
C	1	0.03395
D	1	-0.00345

EXPERIENCE RESIDUALS FOR FULL MODEL

PLOT OF RESIDUAL*YHAT   SYMBOL USED IS *



MATRIX USED FOR MODIFIED GROUP SCREENING DESIGN

OBS	X1	X2	X7	X12	X14	X15	C	D	Y1
1	1	1	1	-1	-1	-1	1	-1	0.685
2	-1	1	1	1	1	-1	1	1	0.631
3	1	-1	-1	-1	-1	1	-1	1	0.701
4	1	1	-1	1	1	-1	-1	-1	0.610
5	-1	1	1	1	-1	1	-1	-1	0.614
6	-1	-1	1	1	1	-1	-1	-1	0.597
7	-1	-1	-1	1	1	1	1	-1	0.708
8	-1	-1	1	-1	1	1	-1	1	0.626
9	1	-1	1	-1	1	1	1	-1	0.741
10	-1	1	-1	1	-1	1	-1	1	0.654
11	1	-1	-1	1	-1	-1	1	-1	0.712
12	-1	1	-1	-1	1	-1	1	1	0.684
13	1	-1	-1	1	1	1	1	1	0.694
14	1	1	1	1	-1	1	1	1	0.693
15	1	1	-1	-1	1	-1	-1	1	0.609
16	1	1	1	-1	1	1	-1	-1	0.635
17	-1	1	-1	-1	-1	1	1	-1	0.742
18	-1	-1	1	-1	-1	-1	1	1	0.697
19	1	-1	1	1	-1	-1	-1	1	0.624
20	-1	-1	-1	-1	-1	-1	-1	-1	0.638

REGRESSION FOR FULL MODEL

DEP VARIABLE: Y1
ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	7	0.03763795	0.00537685	27.766	0.0001
ERROR	12	0.0023238	0.00019365		
C TOTAL	19	0.03996175			

ROOT MSE	0.01391582	R-SQUARE	0.9418
DEP MEAN	0.66475	ADJ R-SQ	0.9079
C.V.	2.093391		

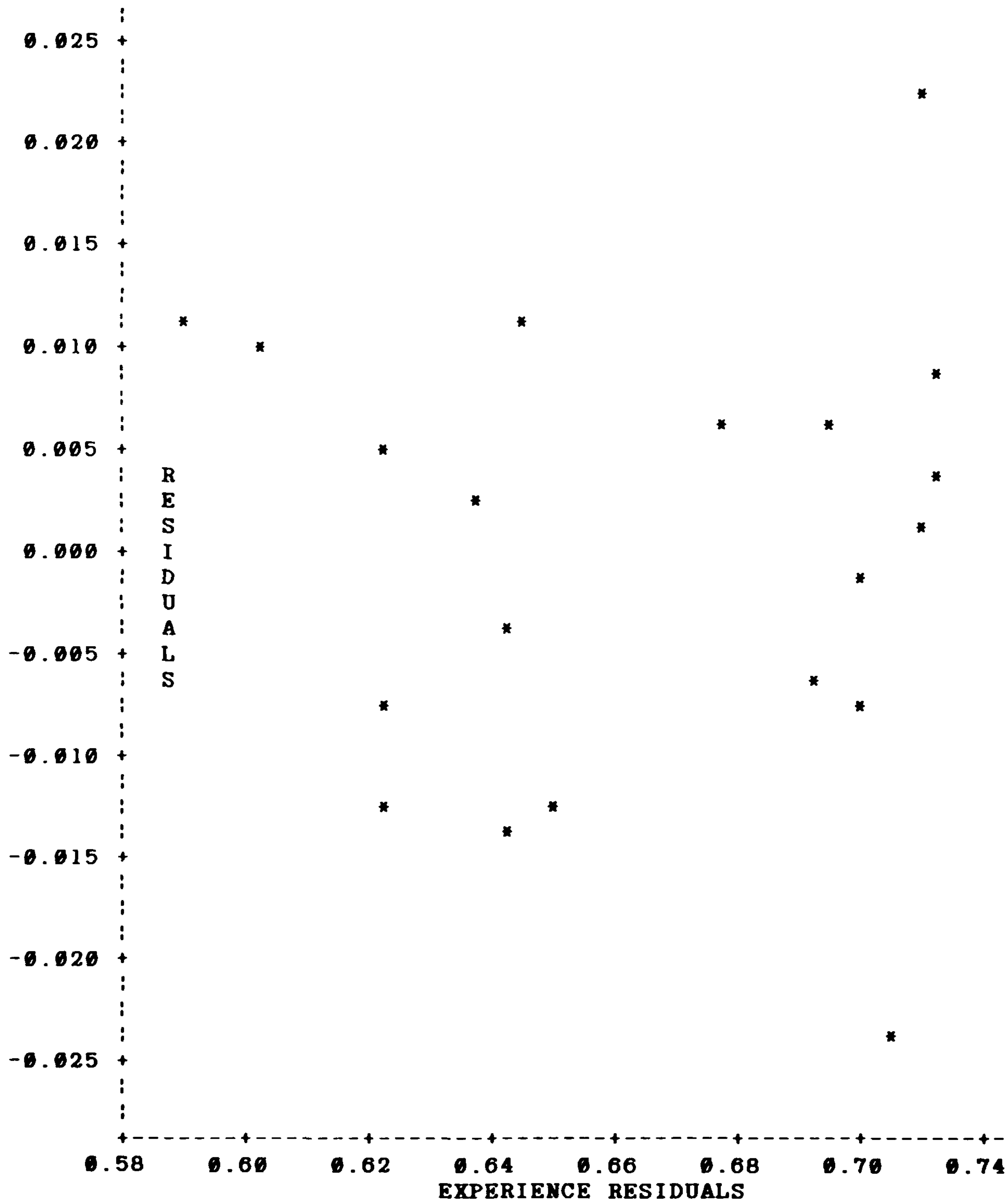
PARAMETER ESTIMATES

VARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR	T FOR H0: PARAMETER=0	PROB > |T|
INTERCEP	1	0.66475000	0.003111672	213.631	0.0001
X1	1	0.00565	0.003111672	1.816	0.0945
X2	1	-0.00905	0.003111672	-2.908	0.0131
X7	1	-0.01045	0.003111672	-3.358	0.0057
X12	1	-0.01105	0.003111672	-3.551	0.0040
X14	1	-0.01125	0.003111672	-3.615	0.0035
X15	1	0.01605	0.003111672	5.158	0.0002
C	1	0.03395	0.003111672	10.911	0.0001

SUM OF RESIDUALS	-2.35922E-16
SUM OF SQUARED RESIDUALS	0.0023238

EXPERIENCE RESIDUALS
FOR MODIFIED GROUP DESIGN

PLOT OF RESIDUAL*YHAT   SYMBOL USED IS *



SUMMARY OF STEPWISE REGRESSION PROCEDURE
FOR INDEPENDENT VARIABLE Y1
OF MODIFIED MODEL

STEP	VARIABLE ENTERED	VARIABLE REMOVED	NUMBER IN	PARTIAL R**2	MODEL R**2	C(P)
1	C		1	0.5769	0.5769	71.3209
2	X15		2	0.1289	0.7058	46.7160
3	X14		3	0.0633	0.7691	35.6447
4	X12		4	0.0611	0.8302	25.0341
5	X7		5	0.0547	0.8849	15.7557
6	X2		6	0.0410	0.9259	9.2969
7	X1		7	0.0160	0.9418	8.0000

STEP	VARIABLE ENTERED	VARIABLE REMOVED	F	PROB>F
1	C		24.5384	0.0001
2	X15		7.4492	0.0143
3	X14		4.3896	0.0524
4	X12		5.3993	0.0346
5	X7		6.6467	0.0219
6	X2		7.1887	0.0189

MATRIX USED FOR TO ESTIMATE MAIN EFFECTS
AND INTERACTIONS

OBS	X1	X2	X3	Y1	X12	X13	X23	X123
1	-1	-1	-1	0.638	1	1	1	-1
2	1	-1	-1	0.612	-1	-1	1	1
3	-1	1	-1	0.696	-1	1	-1	1
4	1	1	-1	0.685	1	-1	-1	-1
5	-1	-1	1	0.639	1	-1	-1	1
6	1	-1	1	0.624	-1	1	-1	-1
7	-1	1	1	0.702	-1	-1	1	-1
8	1	1	1	0.692	1	1	1	1
9	0	0	0	0.651	0	0	0	0
10	0	0	1	0.652	0	0	0	0
11	1	1	0	0.691	1	0	0	0
12	-1	-1	0	0.639	1	0	0	0
13	0	-1	-1	0.626	0	0	1	0
14	0	1	-1	0.691	0	0	-1	0
15	0	-1	1	0.631	0	0	-1	0

VAR	N	MEAN	STD DEV	SUM	MINIMUM	MAXIMUM
X1	15	0.0000000	0.84515425	0.000000	-1.0000000	1.0000000
X2	15	-0.0666667	0.96115010	-1.000000	-1.0000000	1.0000000
X3	15	0.0000000	0.92582010	0.000000	-1.0000000	1.0000000
X12	15	0.1333333	0.83380939	2.000000	-1.0000000	1.0000000
X13	15	0.0000000	0.75592895	0.000000	-1.0000000	1.0000000
X23	15	-0.0666667	0.88371510	-1.000000	-1.0000000	1.0000000
X123	15	0.0000000	0.75592895	0.000000	-1.0000000	1.0000000
Y1	15	0.6579333	0.03125350	9.869000	0.6120000	0.7020000
Y2	15	1.8613333	0.48517842	27.920000	1.1300000	2.5400000
Y3	15	2.5600000	0.61928530	38.400000	1.4500000	3.9800000

MODEL Y1 = B(0) + B(2)*X2

DEP VARIABLE: Y1
ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	1	0.01274087	0.01274087	177.323	0.0001
ERROR	13	0.000934067	.00007185131		
C TOTAL	14	0.01367493			

ROOT MSE	0.008476515	R-SQUARE	0.9317
DEP MEAN	0.6579333	ADJ R-SQ	0.9264
C.V.	1.288355		

PARAMETER ESTIMATES

VARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR	T FOR H0: PARAMETER=0	PROB > :T:
INTERCEP	1	0.66002577	0.00219426	300.796	0.0001
X2	1	0.0313866	0.002357014	13.316	0.0001

COVARIANCE OF ESTIMATES

COVB	INTERCEP	X2
INTERCEP	.00000481478	3.70368E-07
X2	3.70368E-07	.00000555551

SUM OF RESIDUALS	1.38778E-17
SUM OF SQUARED RESIDUALS	0.000934067

MODEL Y1 = B(0) + B(2)*X2 + B(1)*X1

DEP VARIABLE: Y1

ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	2	0.01328736	0.006643681	205.702	0.0001
ERROR	12	0.0003875713	.00003229761		
C TOTAL	14	0.01367493			

ROOT MSE	0.005683098	R-SQUARE	0.9717
DEP MEAN	0.6579333	ADJ R-SQ	0.9669
C.V.	0.8637803		

PARAMETER ESTIMATES

PARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR PARAMETER ESTIMATES	T FOR H0: PARAMETER=0	PROB>:T:
INTERCEP	1	0.66010319	0.001471267	448.663	0.0001
X2	1	0.03254787	0.001605284	20.275	0.0001
X1	1	-0.00750957	0.001825606	-4.113	0.0014

COVARIANCE OF ESTIMATES

COVB	INTERCEP	X2	X1
INTERCEP	.00000216463	1.71796E-07	-3.43592E-08
X2	1.71796E-07	.00000257694	-5.15387E-07
X1	3.43592E-08	-5.15387E-07	.00000333284

SUM OF RESIDUALS	2.77556E-17
SUM OF SQUARED RESIDUALS	0.0003875713

MODEL Y1 = B(0) + B(2)*X2 +B(1)*X1 +B(3)*X3

DEP VARIABLE: Y1
ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	3	0.01333791	0.004445972	145.113	0.0001
ERROR	11	0.0003370187	.00003063806		
C TOTAL	14	0.01367493			

ROOT MSE	0.005535166	R-SQUARE	0.9754
DEP MEAN	0.6579333	ADJ R-SQ	0.9686
C.V.	0.8412958		

PARAMETER ESTIMATES

VARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR	T FOR H0: PARAMETER=0	PROB>|T|
INTERCEP	1	0.66011415	0.001432995	460.653	0.0001
X2	1	0.03271218	0.001568722	20.853	0.0001
X1	1	-0.00754244	0.001778269	-4.241	0.0014
X3	1	0.002059349	0.001603203	1.285	0.2254

COVARIANCE OF ESTIMATES

COVB	INTERCEP	X2	X1	X3
INTERCEP	.00000205347	1.64059E-07	-3.28118E-08	1.36716E-08
X2	1.64059E-07	.00000246089	-4.92178E-07	2.05074E-07
X1	-3.28118E-08	-4.92178E-07	.00000316224	-4.10148E-08
X3	1.36716E-08	2.05074E-07	-4.10148E-08	.00000257026

SUM OF RESIDUALS	-5.55112E-17
SUM OF SQUARED RESIDUALS	0.0003370187

MODEL Y1 = B(Ø) + B(2)*X2 + B(1)*X1 + B(12)*X12

DEP VARIABLE: Y1

ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	3	Ø.Ø1337857	Ø.ØØ4459524	165.524	Ø.ØØØ1
ERROR	11	Ø.ØØØ29636Ø1	.ØØØØ2694183		
C TOTAL	14	Ø.Ø1367493			

ROOT MSE	Ø.ØØ519Ø552	R-SQUARE	Ø.9783
DEP MEAN	Ø.6579333	ADJ R-SQ	Ø.9724
C.V.	Ø.7889176		

PARAMETER ESTIMATES

VARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR	T FOR HØ: PARAMETER=Ø	PROB>¦T¦
INTERCEP	1	Ø.65969283	Ø.ØØ1362137	484.3Ø7	Ø.ØØØ1
X2	1	Ø.Ø325153	Ø.ØØ1466263	22.176	Ø.ØØØ1
X1	1	-Ø.ØØ75Ø3Ø6	Ø.ØØ1667387	-4.5ØØ	Ø.ØØØ9
X12	1	Ø.ØØ3Ø61434	Ø.ØØ1663851	1.84Ø	Ø.Ø929

COVARIANCE OF ESTIMATES

COVB	INTERCEP	X2	X1	X12
INTERCEP	.ØØØØØ185542	1.47255E-Ø7	-2.94511E-Ø8	-3.71Ø83E-Ø7
X2	1.47255E-Ø7	.ØØØØØ214993	-4.29985E-Ø7	-2.94511E-Ø8
X1	-2.94511E-Ø8	-4.29985E-Ø7	.ØØØØØ278Ø18	5.89Ø21E-Ø9
X12	-3.71Ø83E-Ø7	-2.94511E-Ø8	5.89Ø21E-Ø9	Ø.ØØØØØ27684

SUM OF RESIDUALS	5.55112E-17
SUM OF SQUARED RESIDUALS	Ø.ØØØ29636Ø1

MODEL Y1 = B(0) + B(2)*X2 + B(1)*X1 + B(12)*X12 + B(3)*X3

DEP VARIABLE: Y1

ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	4	0.01342899	0.003357248	136.506	0.0001
ERROR	10	0.0002459415	.00002459415		
C TOTAL	14	0.01367493			

ROOT MSE	0.004959248	R-SQUARE	0.9820
DEP MEAN	0.6579333	ADJ R-SQ	0.9748
C.V.	0.7537615		

PARAMETER ESTIMATES

VARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR	T FOR H0: PARAMETER=0	PROB>:T:
INTERCEP	1	0.65970407	0.00130146	506.895	0.0001
X2	1	0.03267942	0.001405604	23.249	0.0001
X1	1	-0.00753588	0.001593249	-4.730	0.0008
X12	1	0.003059186	0.001589706	1.924	0.0832
X3	1	0.002056618	0.001436396	1.432	0.1827

COVARIANCE OF ESTIMATES

COVB	INTERCEP	X2	X1	X12	X3
INT	0.0000016	1.3532E-07	-2.70647E-08	-3.38760E-07	1.12770E-08
X2	1.3532E-07	.00000197	-3.95145E-07	-2.70647E-08	1.64644E-07
X1	-2.7064E-08	-3.951E-07	.00000253844	5.41294E-09	-3.29287E-08
X12	-3.38E-07	-2.706E-08	5.41294E-09	.00000252717	-2.25539E-09
X3	1.127E-08	1.646E-07	-3.29287E-08	-2.25539E-09	.00000206323

SUM OF RESIDUALS	1.80411E-16
SUM OF SQUARED RESIDUALS	0.0002459415

MODEL Y1 = B(Ø) + B(1)*X1 + B(2)*X2 + B(3)*X3 + B(12)*X12 + B(13)*X13 + B(23)*X23 + B(123)*X123

DEP VARIABLE: Y1

ANALYSIS OF VARIANCE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PROB>F
MODEL	7	Ø.Ø1345972	Ø.ØØ1922817	62.541	Ø.ØØØ1
ERROR	7	Ø.ØØØ2152149	.ØØØØ3074499		
C TOTAL	14	Ø.Ø1367493			

ROOT MSE	Ø.ØØ5544816	R-SQUARE	Ø.9843
DEP MEAN	Ø.6579333	ADJ R-SQ	Ø.9685
C.V.	Ø.8427627		

PARAMETER ESTIMATES

VARIABLE	DF	PARAMETER ESTIMATE	STANDARD ERROR	T FOR HØ: PARAMETER=Ø	PROB>:T:
INTERCEP	1	Ø.6596932Ø	Ø.ØØ146Ø628	451.65Ø	Ø.ØØØ1
X1	1	-Ø.ØØ75332	Ø.ØØ1781649	-4.228	Ø.ØØ39
X2	1	Ø.Ø3266598	Ø.ØØ157935	2Ø.683	Ø.ØØØ1
X3	1	Ø.ØØ2Ø43397	Ø.ØØ1613369	1.267	Ø.2458
X12	1	Ø.ØØ3Ø61359	Ø.ØØ1777593	1.722	Ø.1287
X13	1	Ø.ØØ15	Ø.ØØ196Ø389	Ø.765	Ø.4692
X23	1	-Ø.ØØØ14522	Ø.ØØ1691815	-Ø.Ø86	Ø.934Ø
X123	1	-Ø.ØØ125	Ø.ØØ196Ø389	-Ø.638	Ø.544Ø

COVARIANCE OF ESTIMATES

COVB	INTERCEP	X1	X2	X3
INTER	.ØØØØØ213344	-3.77971E-Ø8	1.88985E-Ø7	3.35974E-Ø8
X1	-3.77971E-Ø8	.ØØØØØ317427	-4.98869E-Ø7	-4.59865E-Ø8
X2	1.88985E-Ø7	-4.98869E-Ø7	.ØØØØØ249435	2.29932E-Ø7
X3	3.35974E-Ø8	-4.59865E-Ø8	2.29932E-Ø7	.ØØØØØ26Ø296
X12	-4.26687E-Ø7	7.55942E-Ø9	-3.77971E-Ø8	-6.71948E-Ø9
X13	Ø	Ø	Ø	Ø
X23	2.14183E-Ø7	-5.29684E-Ø8	2.64842E-Ø7	2.6Ø59ØE-Ø7
X123	Ø	Ø	Ø	Ø

COVB	X12	X13	X23	X123
INTEP	-4.26687E-Ø7	Ø	2.14183E-Ø7	Ø
X1	7.55942E-Ø9	Ø	-5.29684E-Ø8	Ø
X2	-3.77971E-Ø8	Ø	2.64842E-Ø7	Ø
X3	-6.71948E-Ø9	Ø	2.6Ø59ØE-Ø7	Ø
X12	.ØØØØØ315984	Ø	-4.28367E-Ø8	Ø
X13	Ø	.ØØØØØ384312	Ø	Ø
X23	-4.28367E-Ø8	Ø	.ØØØØØ286224	Ø
X123	Ø	Ø	Ø	.ØØØØØ384312

Bibliography

American National Standards Institute. "Data Processing - Open Systems Interconnection - Basic Reference Model," Computer Networks, 5: 81-118 (1981).

Bennett, C.J. "The Overheads of Transnetwork Fragmentation," Computer Networks, 6: 21-36 (1982).

Booth, K. K. V. and D. R. Cox. "Some Systematic Supersaturated Designs," Technometrics, 4: 489-495 (1962).

Box, George E. P. and Norman R. Draper. Emperical Model-Building and Response Surfaces. New York: John Wiley & Sons, Inc, 1987.

Cerf, V.G. and E. Cain. "The DoD Internet Architecture Model," Computer Networks, 7: 307-318 (1983).

Damon, Terry M. Chief, Message Switching Operations Branch. Telephone interview and Personal Correspondence. Defense Communications Agency, Washington DC, 1988.

Defense Communications Agency. Additional Security Requirements for the Next-Generation Packet Switch. Final Report. DCA100-84-C-0030. Falls Church: Computer Science Corporation, 17 February 1987.

Defense Communications Agency. DDN Protocol Handbook Volume One. DOD Military Standard Protocols. Menlo Park: DDN Network Information Center, 1985.

Defense Communications Agency. DDN User Operating Procedures. Task Order 5-85, CDRL #003. Vienna: Telecommunications System Center, 30 September 1986.

De Vere Rosemary and L. David Passmore. The DDN Course. Washington: Defense Data Network Program Management Office DCA, January 1986.

Draper, N. R. and H. Smith. Applied Regression Analysis, Second Edition. New York: John Wiley & Sons, Inc, 1981.

Fidelman, James R. and others. Survivability of the Defense Data Network," Signal: 148-158 (May 1986).

Gerla, Mario and H.W. Chan. "Window Selection in Flow Controlled Networks," 1985 Data Communications: 84-92 (1985).

Headquarters Air Force Communications Command. Air Force Defense Data Network Operations Concept. Scott AFB: AFCC/XPP, 9 December 1985.

Heidi, LTC. and B. Heiden. "Defense Data Network," IEEE Communications Magazine: 61-75 (March 1982).

Hofstetter, Helga and Dietmar Weber. "Traffic Models for Large ISDN - PABX's," Computer Networks and ISDN Systems, 10: 192-202 (1985).

Labetoulle, J. and G. Pujolle. "A Study of Flow Through Virtual Circuits Computer Networks," Computer Networks, 5: 119-126 (1981).

Mandl, Christoph. Applied Network Optimization. New York: Academic Press, 1979.

Mauro, Carl A. "Efficient Identification of Important Factors in Large Scale Simulations," 1986 WSC Winter Simulation Conference: 296-305 (December 1986).

Meditch, J.S. and C.T.A. Lea. "Stability and Throughput in Virtual Time CSMA," Computer Networks and ISDN Systems, Volume 10: 7-18 (1985).

Messiah, Adel. "Disnet BLACKER Domains," MITRE Working Paper WP-87W00377. The MITRE Corporation, McLean, 1987.

McQuillan, John M. and others. "The New Routine Algorithm for the ARPANET," Principles of Communication and Networking Protocals: 253-261 (July 1984).

MITRE Corporation. Defense Data Network System Description. Contract No: F19628-84-C-001. McLean: MITRE Division, January 1984.

MITRE Corporation. Defense Data Network Subscriber Security Guide. Contract No: F19628-82-c-0001. McLean: MITRE Division, November 1983.

Narayanan, N.A. "A Common Transport Based Approach to Network Interconnection," Computer Networks and ISDN Systems, Volume 13: 257-267 (1987).

Patel, Ahmed and Michael Purser. "The Design of a Small Packet-Switching Node," Computer Networks and ISDN Networks, Volume 11: 287-295 (1986).

Plackett, R. L. and J. P. Burman. "The Design of Optimum Multifactor Experiment," Biometrika, 33: 305-325 (1946).

Pirkul, Hansan and Sridhar Narasimhan. "Primary and Secondary Route Selection in Backbone Computer Networks." Ohio State University, Columbus Ohio, May 1988.

Rosenberg, Eric. "A Nonlinear Programming Heuristic for Computing Optimal Link Capacities in a Multi-hour Alternative Routing Communications Network," Operations Research, Volume 35, number 3: 354-364 (May-June 1987).

Schwartz, Mischa and Thomas S. Stern. "Routing Techniques Used in Computer Communications Networks," IEEE Transactions on Communications, Volume COM-28, number 4: 539-552 (April 1980).

Selvaggi, Paul S. "The Department of Defense Data Protocol Standardization Program," IEEE Communications Magazine: 111-118 (March 1982).

Tcha, Dong-wan and Kiyoshi Maruyama. "On the Selection of Primary Paths for a Communication Network," Computer Networks and ISDN Systems, Volume 9: 257-265 (1985)

Tropper, Carl. "Priorities and Performance in Packet-Switching Networks," Computer Networks and ISDN Systems, Volume 12: 89-98 (1987).

Vita

Roger Swope [redacted] He attended Shepherd College in Shepherdstown, W. Virginia, and received a Bachelor in Mathematics and Physics in May 1976. He received a commission in the USAF from Officer Training School, Lackland AFB, Texas in April 1980. After completing the Communications Electronics course at Keesler AFB, Mississippi he was assigned to the 2045 Communications Group, Andrews AFB, Maryland, as the On-Site Programmer for the AUTODIN Switching Center. In 1984, he was assigned to Headquarter Strategic Communication Division and served as a Strategic Communication Engineer in the Tactical Warning Assessment Directorate until his assignment to the School of Engineering, Air Force Institute of Technology, in June 1987.

[redacted]

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution Unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GOR/ENS/88D-21	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION School of Engineering 6b OFFICE SYMBOL (If applicable) AFIT/ENS	7a NAME OF MONITORING ORGANIZATION
6c ADDRESS (City, State, and ZIP Code) Air Force Institute of Technology (AU) Wright-Patterson AFB, Ohio 45463-6583	7b ADDRESS (City, State, and ZIP Code)
8a NAME OF FUNDING/SPONSORING ORGANIZATION DCA 8b OFFICE SYMBOL (If applicable) B651	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code) Defense Communications Agency Washington DC 20305-2000	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)

MODELING THE MERGER OF THE CLASSIFIED NETWORKS OF THE DDN: BLACKER

12 PERSONAL AUTHOR(S)
Roger L. Swope, Capt, USAF

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
MS Thesis	FROM ______ TO ______	1988 December	159

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Networks, Communication Networks, Simulation, Defense Data Network
12	08		

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The purpose of this thesis was twofold; first, to determine the impacts of adding the new E3 device, BLACKER, to a network of the DDN and second, to develop a simulation model that would lay the foundation for DDN network modeling. A SLAM II computer simulation model was developed to simulate two networks of the DDN. Components of the BLACKER system were added to the networks. After BLACKER was installed, the two networks were merged into a single Segment.
Sensitivity analysis was performed on the network's input parameters. Analysis was performed on the model's output parameters (values of the SLAM II Output Report) to determine the impact of the BLACKER system on the performance of the individual networks and the Segment.

Thesis Advisor: Major Kenneth W. Bauer, PhD
Associate Professor of Operational Sciences

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL Kenneth W. Bauer, Major, USAF	22b TELEPHONE (Include Area Code) 513-255-3362 22c OFFICE SYMBOL AFIT/ENS

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE